Cognition-Oriented Quadratic Stabilization of Unknown Nonlinear Systems

- A Data-Driven Quadratic Stability Criterion and its Application -

Von der Fakultät für Ingenieurwissenschaften,
Abteilung Maschinenbau und Verfahrenstechnik
der
Universität Duisburg-Essen
zur Erlangung des akademischen Grades
eines
Doktors der Ingenieurwissenschaften
Dr.-Ing.
genehmigte Dissertation

von

Fan Zhang
aus
Xingyang, China

Gutachter: Univ.-Prof. Dr.-Ing. Dirk Söffker
Univ.-Prof. Dr.-Ing. Jörg Raisch
Tag der mündlichen Prüfung: 10. Oktober 2011

Dedicated to my parents

Yuqing Qin and Xike Zhang

who teach me to love the world.

Acknowledgement

First and foremost, I would like to thank my doctor supervisor Univ.-Prof. Dr.-Ing. Dirk Söffker, for his precious support on this fascinating topic of doctor thesis, for his providing an open and free academic environment for research, and most importantly, but also his invaluable guidance on the development of my world view and personality. Without him, neither would this work have been initiated and finished, nor would I have develop a more objective and individual insight than four years ago.

I am also grateful to Univ.-Prof. Dr.-Ing. Jörg Raisch for his effort being the second supervisor for my thesis. Without his keen scientific questions and helpful advice towards the thesis, there would have been still much more aspects to be improved in the technical content of this thesis.

I thank Univ.-Prof. Dr. rer. nat. Peter C. Müller for our discussion in Berlin about the core concept of this thesis when it was still in stages in germination. This discussion has brightened my insight towards this topic and encouraged me to follow the basic idea and brighten it to a doctor thesis.

I also wish to convey thanks to my committee members in the chair of dynamics and control at University of Duisburg-Essen: Hammoud Al-Joumma, Lou'i Al-Shrouf, Dorra Baccar, Kai-Uwe Dettmann, Gregor Flesch, Xingguang Fu, Dennis Gamrad, Frank Heidtmann, Amir Kazaminia, Marcel Langer, Dr.-Ing Yan Liu, Matthias Marx, Dr.-Ing Markus Özbek, Mahmud-Sami Saadawia, Chunsheng Wei, and Xi Shen for their both academical and personal support; Kurt Thelen for his help in the technical aspect; Dr.-Ing Heinz-Dieter Wend, Doris Schleithoff, Yvonne Vengels and Friederike Kögler for their help with administration. Thanks to all these friends and co-workers, past and present, I have spent so much pleasant time during the past four years.

Special thanks to my parents and Miss Zhiguang Wang, who made all this possible, who have given me love and spiritual support when I was in low, and who are still there whenever I need them.

Duisburg, July 2011 Fan Zhang

Abstract

The focus of this thesis is to introduce cognitive capability into automatic control system designed for stabilization problems. Despite of different interpretations of cognition, the point of view in cognitive science that cognition can be treated as a computational process operating on representational structures is adopted in this contribution. Based on this understanding, this thesis proposes a cognition-oriented stabilization method in accordance with the characteristics of cognitive control systems. With the assumption that the system states are fully measurable and the measurements are free of noise, the proposed method can realize quadratic stabilization of unknown nonlinear discrete-time systems. The proposed stabilization method requires neither the information about the system dynamical structure nor the knowledge about system physical behaviors. All the information necessary for stabilizing the unknown system is gained during the interaction of the controller with the unknown system to be controlled.

The core of this thesis is the data-driven quadratic stability criterion, which is taken as the expert knowledge in the proposed control method. This criterion is based on the geometrical interpretation of quadratic Lyapunov functions and transforms the quadratic stability criterion into the problem of judging emptiness of a polyhedral cone, which is identical to solving a max-min optimization problem. Unlike the traditional model-based stability judgment methods, the proposed criterion avoids the utilization of a mathematical model and utilizes the measured data to judge stability, which enables the controller to evaluate the control performance with respect to quadratic stability and develop situated control input to stabilize the plant.

The cognition-oriented stabilization is realized by integrating the proposed data-driven stability criterion (serving as expert knowledge) and contemporary soft-computing techniques (serving as basic cognition functions) into a framework of control which is developed according to a cognitive architecture. The black-box system identification techniques are utilized in the framework to learn the knowledge of plant dynamics. The control input function is generated by the planning module in the framework when the closed-system dynamics is judged as unstable by searching for a suitable control gain according to certain cost function.

Two simulation examples are shown to test the performance of the proposed control method. The first one is the example of stabilizing a pendulum at its inverted position. The recurrent-neural- network is used here to learn the plant dynamics. The second example is the stabilization of a benchmark nonlinear aeroelastic system, where the radial-basis-function network is used as the learning function. In both cases, the plant dynamics are assumed unknown to the controller and all the system states can be measured without noise.

The simulation results of the pendulum example show that the system controlled by the proposed method has symmetric system responses with respect to symmetric initial conditions, respectively. The simulations of the second example are run

under two different nonlinearities and in comparison with the well-established adaptive feedback linearization control methods. The results show that the proposed method can stabilize the system with two different nonlinearities, while the adaptive feedback linearization can stabilize successfully the system with only one the two nonlinearities, without tuning controller structures or parameters. These simulation results of these two example show that the proposed method possesses successful performance of stabilization and good adaptivity to different nonlinear systems.

The limitations of the finished work exist mainly in the conservativeness of the quadratic stability criterion, the great numerical computation power required by the data-driven stability judgment, and the searching speed of suitable control feedback gain to generate suitable control input, which shall be considered and improved in the future work.

Contents

Nomenclature

Denotations

Special sets and spaces

$\varnothing$	empty set
$\mathbb{R}$	real number space
$\mathbb{R}^n$	n-dimensional real vector space
$\mathbb{R}^{m\times n}$	$m \times n$-dimensional real matrix space
$\mathbb{R}^n_+$, $\mathbb{R}^n_-$	n-dimensional positive, negative real vector space

Vectors and matrices

x	a scalar
$\mathbf{1}$	a row vector with all its entries equal to one
$\boldsymbol{x}$	a column vector
$\mathbf{0}$	a column vector with all its entries equal to zero
$[x_i]$	a column vector with scalar x_i being its i-th components
$\boldsymbol{X}$	a matrix
$\boldsymbol{I}$	an identity matrix
$\boldsymbol{X}^T$	the transpose of matrix $\boldsymbol{X}$
$[\boldsymbol{x}_i]$	a matrix with vector $\boldsymbol{x}_i$ being its i-th column vectors
$\mathbf{diag}[\boldsymbol{x}]$	diagonal matrix with elements of vector $\boldsymbol{x}$ being its diagonal entries

Convex geometry and topology

$\mathcal{C}$	a vector set
$\mathbf{conv}\,\mathcal{C}$	convex hull of the vector set $\mathcal{C}$
$\mathbf{cone}\,\mathcal{C}$	convex conic hull of the vector set $\mathcal{C}$ (V-representation)
$\mathbf{cone}(\boldsymbol{A})$	polyhedral cone defined by matrix $\boldsymbol{A}$ (H-representation)
$\mathbf{cone}\,\mathcal{C}^o$	polar cone of the cone $\mathbf{cone}\,\mathcal{C}$
$\mathbf{cone}\,\mathcal{C}^*$	dual cone of the cone $\mathbf{cone}\,\mathcal{C}$
$\mathcal{H}(\boldsymbol{w}, \boldsymbol{b})$	a hyperplane defined by vector $\boldsymbol{w}$ and vector $\boldsymbol{b}$
$\mathcal{H}^+$, $\mathcal{H}^-$	two halfspaces defined by the hyperplane $\mathcal{H}$
h^-	negative halfspace defined by the hyperplane $\mathcal{H}$
$SO(n, \mathbb{R})$	special orthogonal group

Calculations

$\mathbf{rank}\,(\boldsymbol{A})$	rank of the matrix $\boldsymbol{A}$
$\|\boldsymbol{x}\|$	a norm of $\boldsymbol{x}$
$\|\boldsymbol{x}\|_2$	the second order norm of $\boldsymbol{x}$
$\boldsymbol{x} \odot \boldsymbol{y}$	array multiplication defined by $\boldsymbol{x} \odot \boldsymbol{y} = [x_i\, y_i]$
$\langle \boldsymbol{x},\, \boldsymbol{y} \rangle$	inner product of vector $\boldsymbol{x}$ and vector $\boldsymbol{y}$
$\Delta V(\boldsymbol{x})$	difference calculation of function $V(\boldsymbol{x})$
$\mathbf{dim}\,(\mathbf{cone}\,\mathcal{C})$	dimension of the convex cone $\mathbf{cone}\,\mathcal{C}$

Important abbreviations

AoR	Allocation of Resources
BACT	Benchmark Active Control Technology
BP	Back Propagation
CS	Cognitive Science
CTS	Cognitive Technical System
DQLF	Diagonal Quadratic Lyapunov Function
EA	Elastic Axis
GA	Genetic Algorithm
LCO	Limit Cycle Oscillation
min.	minimize
max.	maximize
MPC	Model Predictive Control
NN	Neural Network
PI-O	Proportional Integral Observer
QLF	Quadratic Lyapunov function
RBF	Radial Basis Function
RNN	Recurrent Neural Network
RTRL	Real-Time Recurrent Learning
SOM	Situation Operator Model
s.t.	subject to
T-S	Takagi-Sugeno

1 Introduction

1.1 Motivation

While the modern control techniques have nowadays made remarkable achievements and brought revolutionary convenience to human life, still, substantial additional efforts are required to build more adaptive controllers that can deal with strong nonlinearities, high uncertainties, and especially, unknown dynamics.

Consequentially, a ceaseless pursuit has been made to extend the adaptivity of control systems for the aforementioned challenging circumstances, which has prompt the development of various kinds of control methods being able to cope with unknown systems, such as black-box system identification and control [SZL+95], intelligent control (also called soft-computing methods) [Zad94], data-driven methods [MR08] and so forth. In fact, the appearance and development of these methods have established a set of new research areas in system automatic control and changed the outlook of modern control technologies in both scientific and industrial applications [DO01].

Nevertheless, despite the importance of these efforts to build more flexible and robust control systems, the adaptability of these approaches falls still behind in comparison with biological systems acting in unknown environments, even though some of current control methods have already possessed to certain extent some characteristics similar to human mental capabilities (e.g. the learning ability of neural networks).

Most of the current control approaches coping with uncertain/unknown systems, no matter designed for stabilization, tracking, or other control problems, focus nearly exclusively on methods of exploring the information of local or global system dynamics; in contrast, how the intervention of control processes is itself brought about is not reasoned by the controller itself. In other words, the current control systems cannot penetrate the meaning of the explored information by itself and is therefore unable to apply this information in a flexible manner to achieve the goal of control.

To put it more clearly, the adaptive ability of contemporary control methodologies falls notably behind biological systems at least in two aspects:

- depth in understanding and handling abstract concepts which are universally applicable for dynamical systems, and
- degrees of freedom of applying the knowledge learned from interaction with environments for the purpose of control.

For example, a pilot can control the aeroplane under the disturbances caused by unknown gust wind, although the airdynamics and the related piloting skills under gust disturbance may be new to the human pilot. On the other hand, this task can not be so well accomplished by a machine with current control technology as a human pilot. The reason for that is because a human pilot not only masters the essential skills and the concepts related to aeroplane operation, but also can analyze the situation of the airplane under the encountered gust and make situated decisions based on these skills and understandings.

The superiority of biological systems within these two aspects should be attributed to their working mechanism of mind. Indeed, both the aspects are covered by the concept of *cognition*, which is the main research object in the field of cognitive science [Tha10] and can be understood basically as the learning and use of knowledge.

Cognitive science is the class of studies about the fundamental principles of cognition and the methods of reproducing cognitive capabilities, emerging from 1950s [MGP60]. Although at least early in Plato's time humans began the attempts to understand the working mechanism of mind and its operations [Ben79], the cognitive science in the modern sense, i.e., the study under the assumption that cognitive process is computational and able to be simulated, is still younger, whose first official major institution was founded by a research program in late 1970s [Mil79]. After more than four decades of development, cognitive science has nowadays been developed into "an interdisciplinary study of mind and intelligence, embracing the fields of philosophy, psychology, artificial intelligence, neuroscience, linguistics, and anthropology" [Tha10].

Cognitive science in the engineering context, sometimes called cognitive engineering [Nor96], concentrates on applying the theoretical research results about cognition into design and construction of machines which can reproduce cognitive processes. As pointed in [BW06, Str98], cognitive capabilities can enhance greatly the adaptability of non-cognitive systems to unknown environments. In fact, in recent years a wide variety of approaches which are designed to build artificial machines with cognitive abilities has been published [SA07, BBW07a, AS08, BBG+11] and some of them have been put into real-life applications [CTN07, SG08, GS10, OHS10].

On the other hand, however, most of these applications appear in the field of automation, such as building humanoid robots, autonomous vehicles and other similar topics. Comparatively, the research of reproducing cognitive abilities in the field of automatic control is still less developed than those in automation field. As a consequence, in order to gain more adaptivity it is quite reasonable to introduce cognition and cognitive processes into automatic control systems.

Motivated by this point of view, this contribution focuses on endowing control systems with cognitive capabilities by borrowing the methodology of building artificial cognitive systems in cognitive science, to develop more adaptive, effective, and efficient controllers with less dependence on human interventions when facing unknown

environments. Furthermore, because the stabilization problem, i.e., to design and implement a feedback control law to render the controlled system stable [Kha02], is such a fundamental problem that it encompasses almost all kinds of modern control strategies, it is taken in consideration as the main problem to be solved in this thesis.

The remaining parts of this chapter are organized as follows: in section 1.2, a historical overview of control fields with respect to cognition is made after a brief introduction to cognition and the hierarchy of cognitive control; in section 1.3, the methodologies and applications of reproducing cognitive capabilities in cognitive science are briefly reviewed; with analog to the methodologies in cognitive science, the basic concept of introducing cognitive capabilities into automatic control to solve stabilization problem is introduced in section 1.4 at the end of this chapter.

1.2 Historical overview of control field vis-à-vis cognition

1.2.1 Cognition and cognitive systems

The term *cognition* was firstly utilized in psychology in the nineteenth century to describe the human mental capabilities. Nowadays cognition has become the principal research object in cognitive science.

Due to the great variety of the research fields covered by cognitive science and the incomplete understanding of the mechanism of mind, there exist indeed different interpretations towards cognition [Str01]. Nevertheless, although some contributions attribute cognition exclusively to living organism, such as the minimalist approaches using cognition to refer to consciousness [Sea90], or the maximum methods acclaiming that "living as a process is a process of cognition" [Bod00], the leading point of view towards cognition takes the fundamental hypothesis that "thinking should be understood in terms of representational structures in the mind and computational procedures that operate on those structures" [Tha10]. Accordingly, cognition can be treated as one type of information processing, which indicates that both the conscious and unconscious behaviors are covered within the scope of cognition.

The first well-formulated definition of cognition under the hypothesis mentioned above appears also in psychology as "the activity of knowing: the acquisition, organization, and use of knowledge" [Nei76]. This definition indicates the distinctive features of cognition: learning and utilization of knowledge. This definition is extended in [SHH$^+$95, Str96] from the computational perspective in a more general sense: cognition is a computational process of information taking place based on the mental representation of knowledge, which contains the functions of perception, learning, interpretation and encoding, motivation control, planning, and problem solving.

Correspondingly, as the behavior of a dynamical system (no matter biological or technical) can be considered as the reactions of an agent with respect to the stimuli from its external environment, cognitive systems can be characterized by using the definition of cognition mentioned above as the system which can perceive the external stimuli, assimilate information by learning from its perceptions, structure and store the utilizable information as knowledge, envision consequences of different choice of actions, and change autonomously the direct stimuli-reaction coupling according to its goals [SHH+95].

According to the definition of cognitive systems, it can be seen that the fundamental characteristics of cognitive systems contains three aspects:

- a cognitive system must be able to act interactive with its external environment to exchange information;
- a cognitive system must have the capability of representing and storing the knowledge abstracted from its interaction with the external world; and
- a cognitive system must have the ability of utilizing the learned knowledge flexibly and adaptively to make situated reactions to its perceived stimuli.

From the above illustrated discussion it can be concluded that the dynamics of a cognitive system is de facto governed by the computational process from the stimuli to the reaction with the aforementioned characteristics, which implies that cognition processes and cognitive capabilities are independent from the constitution way of a system and can be attributed unbiasedly to not only biological systems like human or high animals, but also technical systems. This point of view towards cognition and cognitive systems present the possibility of building technical systems with cognitive capabilities and is consequentially adopted in engineering community to guide the construction of artificial cognitive systems and gives the basis of this thesis.

1.2.2 Hierarchy of cognitive control

It is pointed out in [MB96] that "well-designed morphology and automatic behavior can produce intelligent behavior if the environmental conditions can be anticipated during the design phase. Where this is not possible, cognitive (planning and reasoning) processes can be employed to respond intelligently to unpredictable environmental changes". This statement shows that for a cognitive system, its cognitive behavior does not take place of the non-cognitive behavior completely, but can exist in parallel with the non-cognitive one in the same system.

The point of view mentioned above is further deepened in [Str98] and used to establish a three-level framework of the action control of a cognitive system, as shown in the figure 1.1. This three-level working mechanism of a cognitive system describes

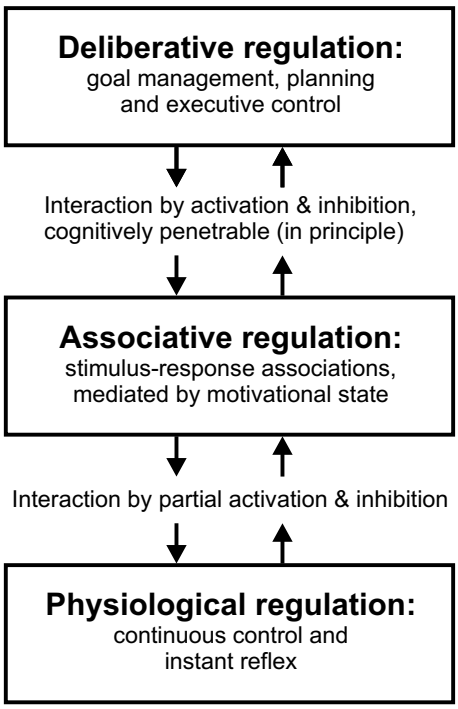


Figure 1.1: Three levels of action control in cognitive systems [Str98]

the regulation hierarchy of a cognitive system's reactions against its external stimuli and can be used to identify whether a system is cognitive or not.

The lowest level within this hierarchy is the physiological regulation, which represents the non-cognitive behavior in a cognitive system. It can be treated simply as the unconscious conditioning reflex action of biological systems. If the stimuli-reaction coupling of a system can be described thoroughly by the lowest level, this system is just a non-cognitive system.

The cognitive behavior is described in the other two levels: the deliberative regulation as the highest level and the associative regulation as the level in between. In the two higher levels, all the computations are taking place based the representation of the knowledge assimilated from the stimuli side, which is one of the most essential features of cognition. However, the manner how the knowledge is utilized is different within the two levels. In the associative regulation level, the knowledge is put into service according to some established working patterns. These working patterns are not produced in the second level, but formulated in the third one by the motivational decision making and strategic planning.

Take a cognitive chess-playing computer as an example. The situations on the chessboard are represented as the learned knowledge; the plans of actions to put the opponent in check is the established working patterns in the second level; but such working patterns are generated by the deliberative regulation. From this example it

can be seen that the differences between those two higher levels are just like those between 'know-how' and 'know-why': in the second level the cognitive system can only know how it should response with the knowledge obtained from external world, but the reason why it should do so can the cognitive system only grab in the third level.

The three levels are not independent from each other but connected by internal interactions. Based on the interactions among these three levels and the interactions between the cognitive system and the external world, the stimuli from the outside world are continuously being processed and knowledge obtained from the stimuli being updated, by which way the cognitive system is able to behave flexibly to adapt the varying unknown external environments.

1.2.3 Review of control fields with respect to cognition

Since the main task of this contribution is to establish a cognition-oriented control method, it is reasonable to perform a retrospective of control methods vis-à-vis their cognitive capabilities. With accordance to the three-level hierarchy of cognitive control, control techniques can be characterized into three classes: 1) control without learning abilities; 2) control with learning abilities and pre-established patterns to fit pre-assumed working contexts; 3) control with learning abilities and self-organized patterns of utilizing the learned information. The overview of the control field with respect to these three classes is shortly given as follows.

Control without learning ability

This first class is the control without learning capabilities, including the analytical control approaches based on mathematical tools, such as the frequency domain methods using Fourier and Laplace transformations, time-domain algebraic approaches, polynomial-matrix-domain approaches, geometric methods, stochastic control, robust control and so forth. A detailed review of these analytical control methods can be found in [FCZ03]. This class of control approaches provides the fundamentals of modern control theory and has made valuable contributions in both the research and the applications of automatic control.

Because this class of methods does not have learning capabilities, the distinguished feature of them is that the controllers designed by these approaches are not able to establish by themselves the models of systems to be controlled (hereafter called plants). After the controllers being designed, they are put in use with no associated mechanism to modify its design in response to the changes of their external environments (changes of the plant dynamics, disturbances, etc). Therefore, instead of adaptivity, the term *low sensitivity* against external changes is used more frequently

to describe the flexibility for this class of controllers. Although these controllers can accommodate certain external changes, this is guaranteed by their 'robust margins' defined in their offline design, rather than online modification of their design by themselves.

From the cognitive point of view, controllers designed by this class of approaches only possess the functionality of the first level within the hierarchy of cognitive control. They cannot build their own representations of the external world, but can only react rigidly to their stimuli (the outputs of the plants) in accordance with pre-established mathematical rule, just like the unconscious conditional reflexes of biological systems.

Control with learning ability and pre-defined working patterns

Online black-box system identification and control [SZL+95, Liu01, Hja05], soft-computing techniques utilizations [Zad94, Ise98, Fod09] including Neural Networks (NN), fuzzy logic, evolutionary methods, and their various combinations between each other, parts of the data-driven control methods which use the information derived from their data-set to describe the plant dynamics [MR08, PI09], and the other control methods which can online establish the model of plant dynamics, belong to the second class: control with learning ability and pre-established patterns suitable for different working contexts. This class of control can realize the functionality of the second level in the hierarchy of cognitive control, due to their online self-built representations of plant dynamics and the fixed manner of using these representations.

In general, this class of control approaches has a specified parameterized controller structure and a corresponding working pattern which incrementally makes adjustments of these parameters, e.g., the adjusting mechanism of model reference adaptive control, by which the representation of the external world (i.e., the dynamics of the plant), no matter local or global, is learned by the controller and kept updating as new situations arrive. The learning capability provides for these controllers of the adaptivity towards the variation of plant dynamics.

It should be mentioned that although pure fuzzy-logic control does not have learning capabilities, it should also be attributed into this class. The fuzzy logic rules can be treated as the partitions of state space of the plant dynamics and each partition as a local model of the plant dynamics. In the local sense, fuzzy logic control must be able to build its own representations of its encountered different situations to match its fuzzy logic model and then follows the established fuzzy logic rules control the plant. From this perspective, the fuzzy-logic-based method is also able to build its own representations of its external environment and utilize them according to some fixed working patterns.

On the other hand, it is necessary to emphasize that the working patterns of this class of control techniques do not change when confronting different environments; nor can they retain the memory of solutions as they have achieved. The reason for this is because these working patterns do not have the ability to *discern* the physical rules that explains why old parameters are changed with new situations. As a result, these working patterns cannot employ these rules self-consciously and thus must rely on their structures to change parameters of the controller. This may lead to the consequences that if the variation of the external environment is beyond its working patterns, the controller would fail to fulfill its control goals, because it does not know the way to adjust its working patterns to adapt the variations.

For example, the NN-based system identification and control is usually realized by the working pattern of model-inverse strategy, i.e., the controller adopts the inverse of an identified model to diminish the plant outputs and replace it with outputs produced by pre-defined goal dynamics. But if the external world is changed and additional performance standards (like the measure of input energy) are required, the same working pattern may probably fail to reach the desired performance.

Control with learning ability and self-organized working patterns

The control with both learning abilities and self-organized working patterns, which covers all the three levels in the hierarchy of cognitive control, is actually cognitive control itself. The learning abilities of this class of control methods realize the mental representation of the outside world; and the self-organization guarantees the deliberative regulation of its working patterns to adapt the unknown.

According to the requirements of the highest level in cognitive control, the class of control approaches with learning ability and self-organized working patterns should have a keen insight about both mental representation of the external world and the interaction mechanism between the cognitive system and the external environment. In the ideal situation, a true cognitive control system should be able to obtain this keen insight from learning by itself. But this learning mechanism is still not clear in the theory of cognitive science. Despite of a few contributions which partially fit the direction of developing this learning mechanism, such as the approximate dynamic programming methods [Len08], the true cognitive control has unfortunately not been completely realized in automatic control.

1.2.4 Supplementary remarks

The review stated above takes some representative examples of modern control approaches to show the fundamental characteristics of the three classes divided according to the three levels of cognitive control, from which it can be seen that the

criterion to attribute a control approach into the aforementioned classifications is the division principles of the three-level hierarchy of cognitive control.

However, it should be noted that due to the various formulations and approaches in control field, even the same type of approaches should be classified specifically, rather than generally. Taking adaptive control approaches for example: on one hand, if the rules of adaption are formulated simply by parameter changes with respect to a specified model without mental representations (e.g. the output linearization adaptive control in [AZSAM10]), this kind of control should be classified into the first class; on the other hand, if the adaption occurs also in the mental representation's level (e.g. the NN-based backstepping control in [ZGH00]), it should be classified into the second class.

1.3 Realizing cognition for engineering applications

1.3.1 Methodologies of reproducing cognitive capabilities

The main engineering applications related to cognitive science are to reproduce the cognitive capabilities within artificial systems. The realization of cognitive capabilities is dependent on three items: the tools to realize basic cognitive functions, the cognitive architecture which determines the mechanism how these functions are internally organized, and a set of knowledge specified for certain application context.

Realizing basic cognitive functions

It is pointed out in [Hol77] that the basic cognitive functions contains *perception*, *interpretation*, *planning*, and *execution*. *Perception* is the function using available knowledge to gather information from stimuli by human sensory activities. Using knowledge from the knowledge or data base, the perceived information is identified, elaborated, and explained by *Interpretation* into new knowledge with the usage of the already acquired and formalized knowledge. *Planning* generates a development of a plan for actions to be carried out by *Execution* which entails the implementation of the decision in the form of responses [Cac98].

These functions are put in service as the basis of a cognitive system and every single one of them can be realized by different technologies, depending on the application circumstances. For example, the planning functions can be realized either by searching the complete solution space, or by probabilistic reasoning. The differences between these two planning methods lie in the efficiency, but not the effect of planning functions in the whole cognitive system. Thus, the realization methods of cognitive functions are changeable with the development of science and technologies.

Cognitive architecture

Cognitive architecture is defined in [Lov01] as "the description of the knowledge representation and the set of mechanism how the knowledge is handled". The research on cognitive architectures has been steadily paid attention to in cognitive science and many different architectures have been proposed, such as Soar [LLR98], ACT-R [ABB+04], ICARUS [LC06], ADAPT [BLL04], SOM [AS08], etc. These cognitive architectures have been implemented to a wide variety of not only oldest mathematical tasks like the 'Tower of Hanoi' or 'maze navigation' but also real life applications. For example, Soar has been used for military operations on land [CTN07] and for the maintenance task of the gas turbine engines on naval ships [RJ00]; ACT-R has been applied in the system to determine the area of anti-air warefare (Anti-Air Warfare Coordinator) [FBD+04] and predictions of activation patterns in brain imaging studies [ABB+04]; ICARUS has been implemented in in-city driving and pole balancing [Lan06]; and so forth.

The internal properties of a cognitive architecture can be characterized by the manner how it handles the knowledge, i.e., the mechanisms of the representation, organization, utilization, and acquisition of knowledge [LLR09]. In general, a cognitive architecture contains the short-term and long-term memories, modules realizing different cognitive functions, and the functional processes that operate on this structures [LLR09]. Based on the organization and the intercommunication among different components, a cognitive architecture can specify the underlying infrastructure of a cognitive system and represents the verity within cognitive processes that are invariant over time and independent from context settings [Lan05].

Despite the different configurations of different cognitive architectures, for example, ACT-R is notably distinguished from Soar because of its special attentions upon motivations, they are not antithetical to each other. In contrast, their shared central issue of a cognitive architecture is the handling of knowledge: acquisition, representation, organization, refinement, and utilization of knowledge [LLR09], which is also the core of the study about cognition. From this perspective, cognitive architectures offers a unified theory of cognition and can be used for both the interpretation of psychological phenomena and the construction of integrated cognitive systems [New90].

Expert knowledge

Cognitive architectures cannot accomplish any tasks by themselves and need to acquire or be provided with knowledge to perform any given task. In general, the knowledge within a cognitive architecture can be classified in three different ways [LLR09]:

- *skill knowledge* about how to generate or execute sequences of actions and *conceptual knowledge* dealing with classes of objects, situations, contexts and so forth [KWM97],
- *semantic knowledge* about depiction of generic concepts and procedures and *episodic knowledge* about specified entities and events encountered in the external environment [Tul72], and
- *expert knowledge* given by human in advance and suitable for some specific tasks and *learned knowledge* which is obtained from the interaction [AS68].

These distinctions are utilized in different architectures respectively to show the different emphasis of the corresponding architectures. For example, the skill-conceptual classification of knowledge is utilized within ICARUS which emphasizes more about the cognitive separation of categories and skills [CTN07].

Indeed, the distinctions above are not mutually exclusive. For example, a piece of expert knowledge can be classified either as skill knowledge, or as conceptual knowledge, depending on its different operational objectives. Nevertheless, in most cases the third division mentioned above is utilized for the sake of storage manner of knowledge.

In the third classification of knowledge, the learned knowledge is used to represent the external world and is mapped into mental representations, while expert knowledge explains the learned knowledge with respect to the goal of actions in a specified context. The expert knowledge is usually organized as long-term memory to store generic skills and concepts that are generic overtime; and the learned knowledge as the short-time memory that contains short lived beliefs and dynamical representation of the external environment.

In fact, the expert knowledge represents the understanding of the interaction mechanism between a system and its environment under a specified context. The ideal case for a cognitive system to obtain expert knowledge should be from learning. As mentioned in section 1.2, the learning mechanism to acquire such kind of abstract knowledge is unfortunately still not clear, which impedes the construction of true cognitive systems in an ideal manner.

Instead, a compromising but useful way is be to incorporate this understanding to the system when it is built. Although systems built in this way cannot learn everything by itself as in the ideal case, it can also adjust by itself its working patterns according to the pre-given expert knowledge and thereby owns to a large extent cognitive capabilities.

According to [Lov01], the combination of cognitive architecture and expert knowledge is defined as cognitive model, which defines the basic rules how cognitive capabilities can be realized in a specified context and is indispensable for any engineering realization of cognitive systems.

1.3.2 Cognitive technical systems

By utilizing the three items explained in the former subsection, cognitive science has been applied into a variety of engineering fields ranging from problem solving tasks in artificial intelligence to command control in human-machine interactions [CTN07]. Among these applications, the construction of Cognitive Technical Systems (CTS) [BBW07b] in automation is of particular interest in this contribution, because the problem settings of cognitive technical systems are overlapped in a significant measure with those of cognitive control systems.

The task of building CTS is to introduce cognitive capabilities into technical systems [BBW07b], which can be realized according to the methodology explained in the former subsection. The well-established examples of CTS may include the humanoid upper body system for two handed manipulation [UBL06], the danger recognition system within a cooperative group of vehicles [BWB09], the mobile robot with learning ability from the interaction with unknown environments using Petri nets [GS10], the driving assisting system constructed according to SOM cognitive architecture [FS10], and etc. From these examples CTS shows its potential of realizing more reliable, flexible, adaptive, and robust behavior than the normal technical systems.

The core of the design and construction of a CTS is the configuration of sensors and actuators integrated in the physical system, and more importantly, the information processing program behind the hardware which performs the task of cognition and makes commands to drive the physical system to act situatedly in the external physical environment. Moreover, due to its cognitive capabilities, CTS requires few pre-given information about the external environment, which can be obtained from the cognition process.

These requirements of the design of CTS are similar to those of designing a cognitive controller which is capable of regulating the dynamics of unknown plants [AS08]. The control law can be taken as an analog of the information processing program in the CTS and the plant as the external environment of the controller. Thus, a cognitive automatic controller should be able to be established according to the similar strategy of constructing CTS, which is the main task of this contribution and is formulated in the next section.

1.4 Problem definition: cognition-oriented stabilization

1.4.1 Problem of cognition-oriented stabilization

As mentioned in the motivation section, the main task of this thesis is to establish a intelligent control method with cognitive capabilities to solve the problem of stabilizing unknown nonlinear dynamical systems. In order to avoid the confusion with

the intelligent control in the traditional sense, the controller proposed in this thesis is given the name of cognition-oriented intelligent control.

Mathematically, the control problem of stabilizing a nonlinear discrete-time system can be formulated as follows: considering the nonlinear discrete-time system

$$\boldsymbol{x}(k+1) = \boldsymbol{f}(\boldsymbol{x}(k), \boldsymbol{u}(k)), \tag{1.1}$$

where $\boldsymbol{f}(\cdot)$ represents the nonlinear system description, $\boldsymbol{x}$ denotes the state vector, $\boldsymbol{u}$ the control input, and k the time, the stabilization problem of the system above is to design a control input

$$\boldsymbol{u}(k) = \boldsymbol{u}(\boldsymbol{x}(k),\, \boldsymbol{x}(k-1),\, ...\,, \boldsymbol{x}(k-l)), \tag{1.2}$$

where l is an integer and $l < k$, such that the origin $\boldsymbol{x} = \boldsymbol{0}$ in the state space of the system (1.1) with control input (1.2) is a uniformly asymptotic stable equilibrium point [Kha02]. It is especially required in this contribution that no structural or physical information of the nonlinear function $\boldsymbol{f}(\cdot)$ described in (1.1) is known, nor the manner how the control input is coupled within the system (1.1) (eg. additive, multiplicative, etc.), but the system states are assumed as fully measurable, so that the proposed controller can fulfill the task of stabilization with only the measurements indistinguishably within any nonlinear systems satisfying the mentioned requirements and to be described as assumed in (1.1).

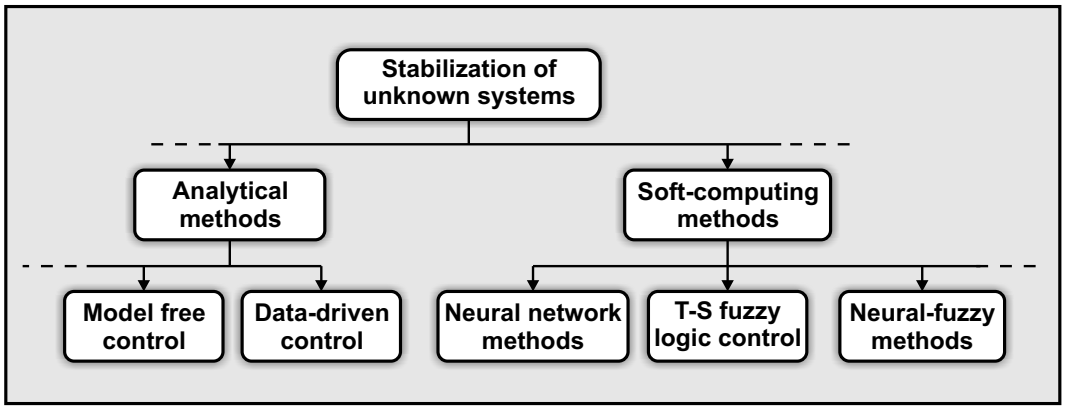


Figure 1.2: Methods of stabilizing unknown systems

The stabilization problem of unknown systems has been widely handled in automatic field and can be solved by different kinds of control methods, such as model-free control [DlSAQ07, FJSR08], NN-based control [Rov99, CHI01, LQZK04, FP10, Kos10], fuzzy-logic control [TC98, JVL00, Rig09], data-driven methods [CCdF99, vHdJS06, PI09], and so forth, which are categorized and shown in figure 1.2. Indeed, satisfactory results can be obtained from these methods and even some of them can already realize some cognitive functions. For example, in the NN-based method used for

stabilization of unknown systems [Rov99, DL05, WOW08], or the T-S fuzzy-logic control of chaotic systems with unknown parameters [KPKP05], or the T-S fuzzy-neural controller [WCL08], the adaptivity can be attributed to the learning capability of neural networks or T-S fuzzy models, while learning ability is one of the fundamental characteristics of cognitive systems.

Nevertheless, the aforementioned approaches follow still some restricted working patterns to rigidly rigidly the expert knowledge in stabilization problem (i.e., the knowledge about stability). For example, the NN-based approaches mentioned above take the inverse model strategy to generate control input, which is a fixed working pattern. Under this circumstance, a stable goal dynamics must be given to the controller in advance, rather than planned by the controller itself. According to the explanations in section 1.2, these control methods are lack of the ability of self-organization of working patterns. Due to this reason, the control systems built according these methods do not belong to the cognitive class, which are less adaptive than cognitive control.

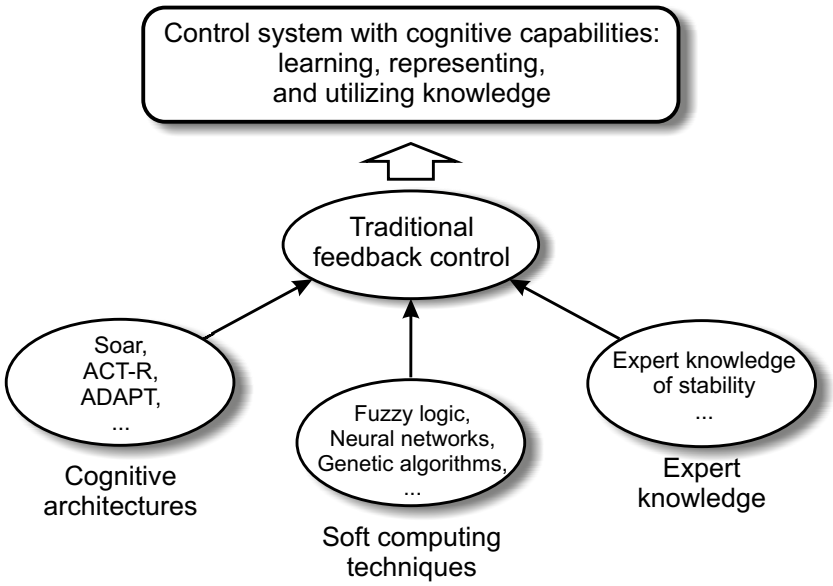


Figure 1.3: Strategy to construct cognition-oriented control systems

From the research and applications of CTS it can be seen that most of the basic cognitive functions can be realized by of soft-computing methods and virtual sensor technologies. Due to this fact and by comparing with the methodologies of constructing cognitive systems, the task of constructing a cognition-oriented intelligent control system requires the other two items: a suitable cognitive architecture and a set of expert knowledge. Thus, a cognition-oriented intelligent controller can be realized by properly arranging these three items together, as shown in figure 1.3.

1.4.2 Requirements towards the expert knowledge

It should be mentioned here the the difficulty of constructing cognition-oriented control in the stabilization problem setting is to find a suitable kind of expert knowledge about stability, as the cognitive architectures for automatic control can be borrowed to a large extent from cognitive science.

In order to realize the cognitive procedures of the deliberative regulation in the three-level hierarchy of cognitive control, the expert knowledge for cognitive stabilization must be able to be *understood* and utilized by the controller itself. In the practical sense, it means that a numerically realizable stability criterion serving as expert knowledge is required for the controller, which should be used not only to judge the stability of the motion of the closed-loop system in real-time, but also to guide the controller to generate suitable control input.

According to the goal of this thesis, it is assumed that the model of the concerned system cannot be given in advance; so a precise model is neither known to the controller nor able to be utilized for the controller design. Therefore, a precise description of the concerned system has to be learned (identified) by the controller.

On the other hand, the dynamics of the closed-loop system with a varying control function, which is necessary for cognitive control to meet its goal, is usually time-variant. Moreover, the identified model describes in most cases only the local dynamics of the plant, which means it cannot be globally accurate and constant over time. Due to these two facts, it can be seen that the stability criterion required by cognition-oriented stabilization cannot be established according to the contemporary model-based stability judgement methods that rely heavily on the system mathematical model.

It is reasonable to believe that the utilization of a mathematical model is not inevitably necessary. Since the identified model used for judging stability shall be obtained from the measured data, the stability of the concerned system should be able to be judged directly from the data set containing system trajectories. From this point of view, the required stability criterion for the proposed method should be able to be fulfilled in a *data-driven* manner, overcoming the shortage of model-based stability criterion caused the inaccuracy and the locality of identified model.

1.4.3 Organization of this thesis

The remaining parts of this thesis are organized as follows: firstly, the expert knowledge about stability utilized in data-driven context is introduced in chapter 2; secondly, the realization of the proposed cognition-oriented stabilization is introduced in chapter 3, including the cognitive framework for stabilization and the realization of each module serving as cognitive function within the framework; thirdly, the

successful performance of proposed method is shown by two numerical examples: the one about the stabilization of an inverse pendulum serving as an introducing example to show detailed realization steps of the proposed method, and the other the stabilization of a nonlinear aeroelastic system as application examples; the last chapter concludes the whole thesis and gives an outlook of the proposed method.

2 Expert Knowledge: A Data-Driven Stability Criterion

As pointed in section 1.4 in chapter 1, a data-driven stability criterion is proposed in this contribution and used as the expert knowledge in the cognition-oriented control approach introduced in this thesis. In this chapter, the proposed data-driven stability criterion is introduced, as well as the corresponding algorithm used for stability judgment.

2.1 Data-driven methods in stability analysis

The term *data-driven* is used to characterize the class of methods that appear in recent years in the field of system analysis and control [MR08]. Unlike the widely used model-based methods which rely to a large extent on a precise mathematical model, the data-driven methods use only measured data of the target system to solve system analysis and control problems, thereby possessing the advantages when a sufficiently precise model is hard to be built.

This feature makes the data-driven stability judgment method more suitable to serve as expert knowledge in cognition-oriented stabilization. In cognition-oriented stabilization, a precise representation of the dynamics of an unknown system cannot be obtained before the control but refined from the interaction between the controller and the plant, which means the representation of the plant dynamics is dynamically changing and should, if necessary, be partitioned into several local models. On the other hand, in model-based stability analysis, the stability can be judged by finding a common Lyapunov function of these different local models, which is usually difficult to be solved algorithmically especially when the local model is nonlinear [DBPL00].

In contrast, because the data-driven method concentrates on the system trajectory, the system dynamics which is contained in the system trajectory, is taken as a whole without being partitioned into local models, by which way the problem of finding a common Lyapunov function is avoided. From this perspective, it can be seen that a data-driven stability criterion is inherently suitable for stability judgment in cognition-based stabilization.

On the other hand, however, most discussions about stability in the data-driven context concentrate on showing certain stability conditions for a specific data-driven control and are difficult to be generalized. For example, in [dBK99] the stability of iterative tuning is proven, and so is the stability of unfalsified control in [vHdJS06]. Nevertheless, these discussions only provide stability conditions to a closed-loop system with one certain control strategy (e.g., iterative tuning, or unfalsified control, etc.), but they can not be used as stability criterions for systems without the corresponding control. Due to this reason, these discussions are difficult to be extended

to assess the stability of an arbitrary dynamical system. An exceptional case is reported in [PI09], where the data-space based stability conditions is proposed in the form of a linear matrix inequality. But it is not shown that the method proposed in [PI09] can be applied to real-time implementation and nonlinear systems. As a consequence, the current data-driven stability analysis method cannot be used directly for online stability judgment of unknown nonlinear systems.

Motivated by these concerns, this contribution proposes a data-driven stability analysis method suitable for quadratic stability assessment of an arbitrary unknown nonlinear system, a quadratic stability condition which can be judged from the measured data of system trajectories. This problem is handled by using of the geometrical links of QLFs with convex cones. The proposed criterion shows that the existence of a QLF can only be guaranteed if the measured system trajectory can be mapped with one certain orthogonal matrix at every time instant into a negative halfspace, which is equivalent to the fact that the corresponding polar cone of the mapped data has a non-empty intersection with the positive real space. It is shown further that the previously mentioned geometric problem can be transformed into a max-min optimization task according to computational geometry theory [Sto73]. Based on this result, an algorithm is developed and can be executed at every time instant to realize an assessment of the system stability, by which the data-driven online stability judgment is realized.

2.2 Problem definition of data-driven stability analysis

The discrete-time nonlinear system concerned in this thesis has the form of

$$\boldsymbol{x}\left(k+1\right)=\boldsymbol{f}(\boldsymbol{x}\left(k\right)), \tag{2.1}$$

with $\boldsymbol{f}(\cdot):\Omega\rightarrow\mathbb{R}^{n}$ a mapping from a compact set $\Omega\subset\mathbb{R}^{n}$ into $\mathbb{R}^{n}$, and with the system state vector $\boldsymbol{x}$ belonging to the region Ω. Following the definition in [Bar85], the quadratic stability for such systems can be stated as follows:

Quadratic Stability The discrete-time nonlinear system (2.1) is defined to be quadratic stable if there exists a positive definite Hermitian matrix $\boldsymbol{P}$ such that the first-order difference of the function $V(\boldsymbol{x}\left(k\right))=\boldsymbol{x}\left(k\right)^{T}\boldsymbol{P}\,\boldsymbol{x}\left(k\right)$ along the solution of system (2.1) satisfies

$$\begin{aligned}\Delta V(\boldsymbol{x}\left(k\right)) &= V(\boldsymbol{x}\left(k+1\right))-V(\boldsymbol{x}\left(k\right))\\ &= V(f(\boldsymbol{x}\left(k\right)))-V(\boldsymbol{x}\left(k\right))\leq 0.\end{aligned} \tag{2.2}$$

Correspondingly, the function $V(\boldsymbol{x}(k)) = \boldsymbol{x}(k)^T \boldsymbol{P} \boldsymbol{x}(k)$ is named as the Quadratic Lyapunov Function (QLF). If in addition $\boldsymbol{P}$ is diagonal, $V(\boldsymbol{x}(k))$ is named as Diagonal Quadratic Lyapunov Function (DQLF) and the related system (2.2) is defined to be diagonally quadratic stable.

In the data-driven context, the existence of a QLF cannot be determined by using the analytical form of $\boldsymbol{f}(\boldsymbol{x})$ because it is unknown. Suppose that the system (2.1) be fully observable and the system states be measured without noise. At the time instant $t = r$, the data set containing r consecutive measurements of system states can be denoted as

$$\mathcal{X}_r = \{\boldsymbol{x}(1), \ldots, \boldsymbol{x}(r)\} \ . \tag{2.3}$$

The task of online stability judgment in this contribution is defined as to determine the existence of a QLF directly from the data set (2.3) instead of a mathematical description of $\boldsymbol{f}(\boldsymbol{x})$ at every time instant. The system is judged as quadratic stable if and only if a QLF can be found based on the measured data.

2.3 Geometrical preliminaries

Before the main discussion, the denotations and some geometric concepts used in this thesis are briefly introduced in this section. Most of the geometrical definitions are taken from [BV04], to which interested readers can refer for more details.

Convex Set A set $\mathcal{C}$ in a real vector space is said to be *convex* if, for any two different vectors $\boldsymbol{x}$ and $\boldsymbol{y}$ in $\mathcal{C}$ and any positive scalar α in the interval $[0,\ 1]$, the point $(1-\alpha)\,\boldsymbol{x} + \alpha \boldsymbol{y}$ is also in $\mathcal{C}$.

Cone and Convex Cone A set $\mathcal{C}$ is called a *cone*, if for every $\boldsymbol{x} \in \mathcal{C}$ and a scala $\theta \geq 0$ the relation $\theta \boldsymbol{x} \in \mathcal{C}$ is true.

Correspondingly, a *convex cone* is a vector set that is both convex and a cone, i.e., the set $\mathcal{C}$ is a convex cone if for any $\boldsymbol{x}_1,\ \boldsymbol{x}_2 \in \mathcal{C}$ and $\theta_1,\ \theta_2 \geq 0$, the relation $\theta_1 \boldsymbol{x}_1 + \theta_2 \boldsymbol{x}_2 \in \mathcal{C}$ is true.

A convex cone is called proper if it is closed, has nonempty interior (solid), and contains no line (pointed).

Convex Hull The convex hull for a vector set $\mathcal{C}$, denoted as $\mathbf{conv}\,\mathcal{C}$, is defined as the set of all convex combinations of vectors in $\mathcal{C}$, i.e.,

$$\mathbf{conv}\,\mathcal{C} = \left\{ \sum_{i=1}^{m} \theta_i \boldsymbol{x}_i | \boldsymbol{x}_i \in \mathcal{C},\, \theta_i \geq 0,\, \sum_{i=1}^{m} \theta_i = 1, i = 1, \ldots, m \right\}. \tag{2.4}$$

The convex hull is the smallest convex set containing the vector set $\mathcal{C}$: if $\mathcal{B}$ is any convex set containing $\mathcal{C}$, then $\mathbf{conv}\,\mathcal{C} \subseteq \mathcal{B}$. It is obvious that a convex hull in the 2-D plane is a polygon, in the three dimension space a polyhedron, and in higher dimensions a polytope.

Convex Conic Hull A convex conic hull of a set $\mathcal{C}$ is the set of all conic combinations of the vectors in $\mathcal{C}$, defined as

$$\mathbf{conv}\,\mathcal{C} = \left\{ \sum_{i=1}^{m} \theta_i \boldsymbol{x}_i | \boldsymbol{x}_i \in \mathcal{C},\, \theta_i \geq 0,\, i = 1,\, ...\, m \right\}. \tag{2.5}$$

Because a convex conic hull must be a cone (actually, it is the smallest convex cone containing the vector set $\mathcal{C}$), it is denoted in this contribution as $\mathbf{cone}\,\mathcal{C}$.

Polyhedron and Polyhedral Cone A set $\mathcal{C}$ is said to be a convex polyhedron if it can be written as

$$\mathbf{conv}\,\mathcal{C} = \{\boldsymbol{x} | \boldsymbol{A}\boldsymbol{x} \geq \boldsymbol{b}\}, \tag{2.6}$$

with respect to some matrix $\boldsymbol{A}$ and vector $\boldsymbol{b}$. A set $\mathcal{C}$ is a polyhedral cone if it can be represented by the above form of a polyhedron with $\boldsymbol{b} = \boldsymbol{0}$.

A convex polyhedron, or a polyhedral cone, can be considered as the set of solutions to a finite system of inequalities. According to the Minkowski and Weyl theorem [Zie97], a convex conic hull is polyhedral if and only if it is generated by a finite number of vectors, i.e.,

$$\mathbf{cone}\,\mathcal{C} = \left\{ \sum_{i=1}^{m} \theta_i \boldsymbol{x}_i | \boldsymbol{x}_i \in \mathcal{C},\, \theta_i \geq 0,\, i = 1,\, ...\, m \right\} \tag{2.7}$$

is polyhedral if $m \neq \infty$.

There are two representations of a polyhedral cone: the H-representation utilizing the form of the set of inequalities with respect to a matrix $\boldsymbol{A}$, denoted as $\mathbf{cone}(\boldsymbol{A})$ in this work, and the V-representation utilizing the conic combination of the vectors within a set $\mathcal{C}$, denoted in this work as $\mathbf{cone}\,\mathcal{C}$. The Carathéodory theorem [Car11] shows that if the set $\mathcal{C}_p$ contains the extreme rays of the cone defined by $\mathcal{C}$, then the polyhedral cone $\mathbf{cone}\,\mathcal{C}_p$ is identical the the polyhedral cone $\mathbf{cone}\,\mathcal{C}$.

For example, suppose that the vector set $\mathcal{C}$ contain three elements: $\boldsymbol{\eta}_1 = [1,\ 2]^T$, $\boldsymbol{\eta}_2 = [1,\ 1]^T$ and $\boldsymbol{\eta}_3 = [2,\ 1]^T$. The polyhedral cone determined by this vector set is shown as the shaded area in the figure 2.1. This polyhedral cone can be described by the H-representation as

$$\mathbf{cone}\,\mathcal{C} = \left\{ \boldsymbol{x} \in \mathbb{R}^2 \,|\, \boldsymbol{A}\,\boldsymbol{x} \geq \boldsymbol{0} \right\} \text{ with } \boldsymbol{A} = \begin{bmatrix} 2 & -1 \\ -1 & 2 \end{bmatrix}. \tag{2.8}$$

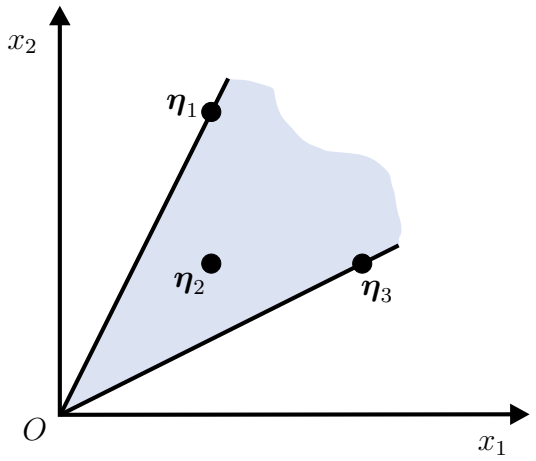


Figure 2.1: An example of polyhedral cone

Denote the extreme rays in $\mathcal{C}$ as $\mathcal{C}_{\text{in}}$. Apparently the elements of $\mathcal{C}_{\text{in}}$ are $\boldsymbol{\eta}_1$ and $\boldsymbol{\eta}_3$. Hence, the V-representation of the polyhedral cone in figure 2.1 can be written as

$$\mathbf{cone}\,\mathcal{C}_{\text{in}} = \left\{ \sum_{i=1}^{2} \theta_i \boldsymbol{x}_i | \boldsymbol{x}_i \in \mathcal{C}_{\text{in}},\, \theta_i \geq 0,\, i = 1,\, 2 \right\} \text{ with } \mathcal{C}_{\text{in}} = \{\boldsymbol{\eta}_1,\, \boldsymbol{\eta}_3\}\,. \quad (2.9)$$

Another typical example of a polyhedral cone is the positive real space $\mathbb{R}^n_+$, whose H-representation is $\mathbb{R}^n_+ = \mathbf{cone}(\boldsymbol{I}) = \{\boldsymbol{x} | \boldsymbol{I}\boldsymbol{x} \geq \boldsymbol{0}\}$, where $\boldsymbol{I}$ is the $n \times n$ identity matrix.

Dual Cone and Polar Cone The polar cone of a convex cone $\mathbf{cone}\,\mathcal{C}$ determined by the vector set $\mathcal{C}$ is denoted as $\mathbf{cone}\,\mathcal{C}^o$ and defined as

$$\mathbf{cone}\,\mathcal{C}^o = \left\{ \boldsymbol{y} | \boldsymbol{x}^T\boldsymbol{y} \leq 0, \text{for all } \boldsymbol{x} \in \mathbf{cone}\,\mathcal{C} \right\}. \quad (2.10)$$

The dual cone of a polyhedral cone $\mathbf{cone}\,\mathcal{C}$, denoted as $\mathbf{cone}\,\mathcal{C}^*$, is defined as

$$\mathbf{cone}\,\mathcal{C}^* = \left\{ \boldsymbol{y} | \boldsymbol{x}^T\boldsymbol{y} \geq 0, \text{for all } \boldsymbol{x} \in \mathbf{cone}\,\mathcal{C} \right\}. \quad (2.11)$$

The relationship among $\mathbf{cone}\,\mathcal{C}$ and its dual cone and polar cones are graphically shown in figure 2.2. If $\mathbf{cone}\,\mathcal{C}$ is a convex cone, then its polar cone and dual cone are also convex. The polar cone contains all the vectors that have negative inner products with the vectors of $\mathbf{cone}\,\mathcal{C}$; and the dual cone the positive inner products with elements of $\mathbf{cone}\,\mathcal{C}$. Furthermore, an important characteristics between the polar cone $\mathbf{cone}\,\mathcal{C}^o$ and the dual cone $\mathbf{cone}\,\mathcal{C}^*$ with respect to $\mathbf{cone}\,\mathcal{C}$ is that $\mathbf{cone}\,\mathcal{C}^o = -\mathbf{cone}\,\mathcal{C}^*$, as shown in figure 2.2.

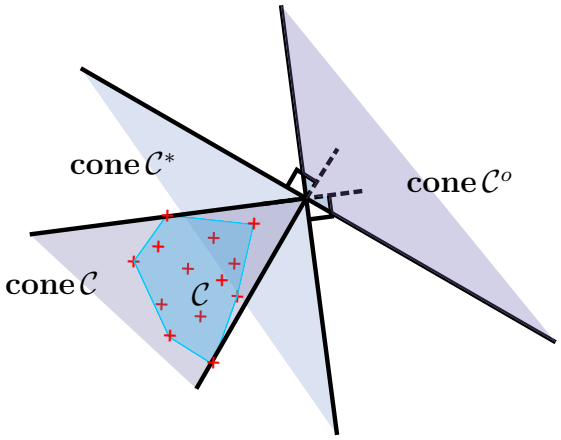


Figure 2.2: Dual cone **cone** $\mathcal{C}^*$ and polar cone **cone** $\mathcal{C}^o$ with respect **cone** $\mathcal{C}$

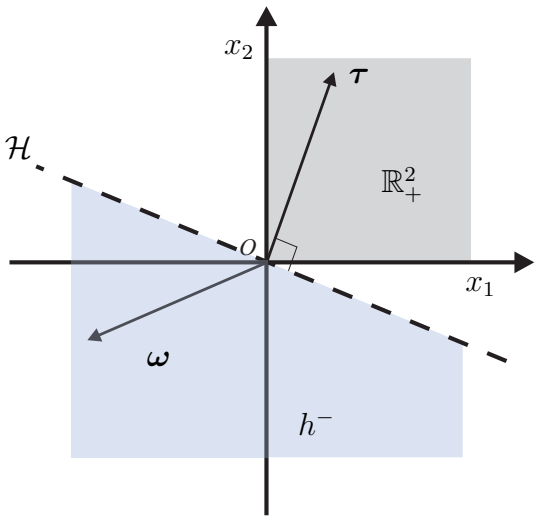


Figure 2.3: A negative halfspace in $\mathbb{R}^2$

Hyperplane and Negative Halfspace A hyperplane $\mathcal{H}(\boldsymbol{w}, \boldsymbol{b})$ in the n-dimensional space is the set defined by the vector $\boldsymbol{w} \in \mathbb{R}^n$ and $\boldsymbol{w} \neq \mathbf{0}$ as

$$\mathcal{H}(\boldsymbol{w}, \boldsymbol{b}) = \{\boldsymbol{x} | \langle \boldsymbol{w}, \boldsymbol{x} \rangle + b = 0\}, \tag{2.12}$$

where $\langle \cdot \rangle$ represents the inner product, the vector $\boldsymbol{w} \in \mathbb{R}^n$, $\boldsymbol{w} \neq \mathbf{0}$ denotes the normal vector of the hyperplane, and $b \in \mathbb{R}$ is a scalar.

A hyperplane divides $\mathbb{R}^n$ into two halfspaces, which are defined as $\mathcal{H}^+ = \{\boldsymbol{x} | \langle \boldsymbol{w}, \boldsymbol{x} \rangle + b \geq 0\}$ and $\mathcal{H}^- = \{\boldsymbol{x} | \langle \boldsymbol{w}, \boldsymbol{x} \rangle + b \leq 0\}$ respectively. Especially, if $b = 0$ and $\boldsymbol{w} > \mathbf{0}$, the relation $\mathbb{R}^n_- \subseteq \mathcal{H}^-$ holds, i.e., the negative real space is a subspace of the halfspace $\mathcal{H}^-$, then the halfspace $\mathcal{H}^-$ is called the negative halfspace, denoted as h^-.

A negative halfspace h^- in the two dimensional space is shown in figure 2.3, where $\mathcal{H}$ is its respective non-vertical hyperplane. Clearly it can be seen that if a vector

$\boldsymbol{\omega}$ is located in a negative halfspace h^-, there must be at least one vector $\boldsymbol{\tau} \in \mathbb{R}^n_+$ such that $\langle \boldsymbol{\omega}, \boldsymbol{\tau} \rangle \leq 0$.

In other words, a negative halfspace h^- is a complete set containing all the vectors $\boldsymbol{\omega}$ that have non-positive inner products with at least one arbitrary vector $\boldsymbol{\tau}$ located in the space $\mathbb{R}^n_+$.

2.4 Data-driven quadratic stability judgement

2.4.1 Relations between DQLF and QLF

Searching a QLF $V(\boldsymbol{x}) = \boldsymbol{x}(k)^T \boldsymbol{P} \boldsymbol{x}(k)$ is equal to searching a suitable positive definite matrix $\boldsymbol{P}$. In [CGH03], it is shown that the complete set of the matrix $\boldsymbol{P}$ in QLF can be mapped to surjectively from the special orthogonal group $SO(n, \mathbb{R})$ and the conventional topology of $\mathbb{R}^n_+$. This mapping can be defined as

$$(\boldsymbol{\Phi}, \boldsymbol{d}) \mapsto \boldsymbol{P} : \boldsymbol{P} = \boldsymbol{\Phi}^T \mathbf{diag}\,[\boldsymbol{d}] \boldsymbol{\Phi}, \tag{2.13}$$

where $\boldsymbol{\Phi}$ is an orthogonal matrix in $SO(n, \mathbb{R})$ and $\boldsymbol{d}$ is a real vector in $\mathbb{R}^n_+$. Because the mapping (2.13) is surjective, which is proven in [CGH03], it can be concluded that no QLF exists if no element over the complete set $SO(n, \mathbb{R}) \times \mathbb{R}^n_+$ can be found and to construct a QLF, and vice versa. Therefore, the existence of a QLF can be determined by searching through the special orthogonal group $SO(n, \mathbb{R})$ and the conventional topology of $\mathbb{R}^n_+$.

By left-multiplying an orthogonal matrix $\boldsymbol{\Phi}$ given in (2.13) to the both side of the concerned discrete-time system (2.1), $\boldsymbol{x}(k+1) = \boldsymbol{f}(\boldsymbol{x}(k))$, a transformed system can be obtained as

$$\boldsymbol{z}(k+1) = \boldsymbol{g}(\boldsymbol{z}(k)), \tag{2.14}$$

with $\boldsymbol{z}(k) = \boldsymbol{\Phi}\,\boldsymbol{x}(k)$ and $\boldsymbol{g}(\boldsymbol{z}(k)) = \boldsymbol{\Phi}\,\boldsymbol{f}(\boldsymbol{x}(k))$.

If the discrete-time system (2.1) has a QLF $V(\boldsymbol{x}) = \boldsymbol{x}(k)^T \boldsymbol{P} \boldsymbol{x}(k)$, according to the definition of QLF it can be obtained that $\Delta V(\boldsymbol{x}) = \boldsymbol{x}(k+1)^T \boldsymbol{P} \boldsymbol{x}(k+1) - \boldsymbol{x}(k)^T \boldsymbol{P} \boldsymbol{x}(k) < 0$. Correspondingly, taking the function $V_z(\boldsymbol{z}) = \boldsymbol{z}(k)^T \boldsymbol{D} \boldsymbol{z}(k)$ as the Lyapunov function candidate for the transformed system, with $\boldsymbol{D}$ being the diagonal matrix $\mathbf{diag}\,[\boldsymbol{d}]$ given in the mapping (2.13), it can be deduced that

$$\begin{aligned} \Delta V_z(\boldsymbol{z}) &= \boldsymbol{z}(k+1)^T \boldsymbol{D}\, \boldsymbol{z}(k+1) - \boldsymbol{z}(k)^T \boldsymbol{d}\, \boldsymbol{z}(k) \\ &= \boldsymbol{z}(k+1)^T \boldsymbol{\Phi}^T \boldsymbol{D}\, \boldsymbol{\Phi} \boldsymbol{z}(k+1) - \boldsymbol{z}(k)^T \boldsymbol{\Phi}^T \boldsymbol{d}\, \boldsymbol{\Phi} \boldsymbol{z}(k) \\ &= \boldsymbol{x}(k+1)^T \boldsymbol{P}\, \boldsymbol{x}(k+1) - \boldsymbol{x}(k)^T \boldsymbol{P}\, \boldsymbol{x}(k) < 0, \end{aligned} \tag{2.15}$$

which shows that the function $V_z(\boldsymbol{z})$ is a DQLF for the transformed system (2.14). On the other hand, if the transformed system (2.14) had a DQLF, it can be proven similarly to the above discussion that the system (2.1) has a QLF, which is composed by the orthogonal matrix in the system transformation and the diagonal matrix in the DQLF of system (2.14). From this discussion it can be seen that a QLF of the system (2.1) is equivalent to the DQLF of its corresponding transformed system (2.14), which is stated as the following lemma,

Lemma 2.4.1 *If the nonlinear discrete-time system (2.1) has a QLF* $V(x\,(k)) = x\,(k)^T\,\boldsymbol{P}\,x\,(k)$*, then there exists an orthogonal matrix* $\boldsymbol{\Phi}$ *such that the transformed system (2.14) possesses a DQLF as* $V_d(\boldsymbol{z}\,(k)) = \boldsymbol{z}\,(k)^T\,\boldsymbol{D}\,\boldsymbol{z}\,(k)$*, where* $\boldsymbol{D} = \boldsymbol{\Phi}\,\boldsymbol{P}\,\boldsymbol{\Phi}^T$ *and* $\boldsymbol{z}\,(k) = \boldsymbol{\Phi}\,\boldsymbol{x}\,(k)$*, and vice versa.*

Furthermore, it can be seen from the discussion above that the orthogonal matrix $\boldsymbol{\Phi}$ used in the system transformation is exactly the same as the orthogonal matrix used in the mapping (2.13). Recalling the conclusion taken from [CGH03] that searching a QLF is equivalent to searching all through the combinations in the special orthogonal group $SO(n,\mathbb{R})$ and the conventional topology of $\mathbb{R}^n_+$, it can be concluded that to search a QLF within $SO(n,\mathbb{R}) \times \mathbb{R}^n_+$ for the system (2.1) is equivalent to searching a DQLF within $\mathbb{R}^n_+$ for the system (2.14) transformed with every element in the special orthogonal group $SO(n,\mathbb{R})$.

This fact not only shows that the lemma 2.4.1 is both necessary and sufficient, but also provides us an idea to determine the existence of a QLF for a discrete-time nonlinear system: if no DQLF exists for every possible orthogonal transformation of the concerned system, the concerned system has no QLF and is correspondingly not quadratic stable.

2.4.2 Necessary and sufficient condition for existence of QLF

Define a transformation for every vector $\boldsymbol{x}(k) \in \Omega$ as

$$\boldsymbol{v}(k) = \boldsymbol{x}(k+1) \odot \boldsymbol{x}(k+1) - \boldsymbol{x}(k) \odot \boldsymbol{x}(k)\,, \tag{2.16}$$

where $\boldsymbol{v}(k)$ represents the corresponding transformed vector and the symbol $\odot$ represents an array multiplication defined by

$$\boldsymbol{a} \odot \boldsymbol{b} = [a_j\,b_j],\ j = 1,\,...\,,\,n. \tag{2.17}$$

In (2.17), the symbols $\boldsymbol{a}$ and $\boldsymbol{b}$ represent two arbitrary n-dimensional vectors with a_j and b_j being their components, respectively. Unlike the inner products between two vectors, the calculation $\odot$ establish a manipulation from two vectors to a new vector.

Applying the proposed vector manipulation to all the elements within the data set $\mathcal{X}$ defined in equation (2.3), a new vector set of $\boldsymbol{v}(k)$ can be obtained. Denote the complete vector set of $\boldsymbol{v}(k)$, $k = 1\,, \ldots, \infty$, as $\mathcal{V}$, and the convex conic hull (the smallest convex cone) determined by $\mathcal{V}$ as $\mathbf{cone}\,\mathcal{V}$.

Consider the nonlinear discrete-time system (2.1) with an equilibrium point at the origin of the state space. The necessary and sufficient condition of existence of a DQLF can be given as the following theorem:

Theorem 2.4.2 *There exists a DQLF $V_d(\boldsymbol{x})$ for the considered nonlinear discrete-time system (2.1) within the domain Ω if and only if for all the $\boldsymbol{x}(k) \in \Omega$, the convex conic hull $\mathbf{cone}\,\mathcal{V}$ of the transformed vectors $\boldsymbol{v}(k)$ is located in a negative halfspace H^- of $\mathbb{R}^n$.*

Proof: To prove sufficiency, suppose the convex conic hull $\mathbf{cone}\,\mathcal{V}$ lie in a negative halfspace h^- of $\mathbb{R}^n$. Obviously all the vectors $\boldsymbol{v}$ within the set $\mathbf{cone}\,\mathcal{V}$ also belong to h^-, because $\mathbf{cone}\,\mathcal{V} \subseteq h^- \subset \mathbb{R}^n$.

Thus, according to the definition of the negative halfspace, there must exist at least one vector located in $\mathbb{R}^n_+$, denoted as $\boldsymbol{d}$ and $\boldsymbol{d} \in \mathbb{R}^n_+$, which has non-positive inner products with any vector $\boldsymbol{v}(k)$ belonging to $\mathbf{cone}\,\mathcal{V}$, i.e.,

$$\langle \boldsymbol{v}(k), \boldsymbol{d} \rangle = \boldsymbol{v}(k)^T\,\boldsymbol{d} = \boldsymbol{d}^T \boldsymbol{v}(k) \;\leq\; 0,\; \boldsymbol{d} \in \mathbb{R}^n_+. \tag{2.18}$$

Using the definition of $\boldsymbol{v}(k)$ in (2.16), the inner product between $\boldsymbol{v}(k)$ and $\boldsymbol{d}$ can be represented as

$$\langle \boldsymbol{v}(k), \boldsymbol{d} \rangle = \boldsymbol{d}^T\;(\boldsymbol{x}(k+1) \odot \boldsymbol{x}(k+1) - \boldsymbol{x}(k) \odot \boldsymbol{x}(k))\,. \tag{2.19}$$

Define a diagonal matrix $\boldsymbol{D}$ as $\boldsymbol{D} = \mathbf{diag}\,[\boldsymbol{d}]$. Obviously $\boldsymbol{D}$ is positive definite because it is diagonal and its diagonal elements vector $\boldsymbol{d}$ belongs to $\mathbb{R}^n_+$. With notation that

$$\begin{aligned} \boldsymbol{x}(k+1) \odot \boldsymbol{x}(k+1) &= \mathbf{diag}\,[\boldsymbol{x}(k+1)]\boldsymbol{x}(k+1)\,, \\ \boldsymbol{d}^T \mathbf{diag}[\boldsymbol{x}(k+1)] &= \boldsymbol{x}(k+1)^T\,\mathbf{diag}[\boldsymbol{d}]\,, \end{aligned} \tag{2.20}$$

and the similar relations for $\boldsymbol{d}$ and $\boldsymbol{x}(k)$, one can obtain the following equation by substituting (2.19) and (2.20) into the inequality (2.18), as

$$\begin{aligned} \langle \boldsymbol{v}(k), \boldsymbol{d} \rangle &= \boldsymbol{x}(k+1)^T\,\mathbf{diag}\,[\boldsymbol{d}]\,\boldsymbol{x}(k+1) - \boldsymbol{x}(k)^T\,\mathbf{diag}\,[\boldsymbol{d}]\,\boldsymbol{x}(k) \\ &= \boldsymbol{x}(k+1)^T\,\boldsymbol{D}\,\boldsymbol{x}(k+1) - \boldsymbol{x}(k)^T\,\boldsymbol{D}\,\boldsymbol{x}(k) \leq 0\,. \end{aligned} \tag{2.21}$$

According to the definition of DQLF it can be seen that the function $V_d(\boldsymbol{x}(k)) = \boldsymbol{x}(k)^T \boldsymbol{D}\, \boldsymbol{x}(k)$ is a DQLF for the concerned nonlinear discrete-time system, because $\boldsymbol{D}$ is a diagonal positive definite matrix and $\Delta V_d(\boldsymbol{x}(k)) = \boldsymbol{x}(k+1)^T \boldsymbol{D}\, \boldsymbol{x}(k+1) - \boldsymbol{x}(k)^T \boldsymbol{D}\, \boldsymbol{x}(k) < 0$. This proves the sufficiency of the proposed theorem.

To prove the necessity, suppose there exist a diagonal quadratic Lyapunov function within the domain Ω, denoted as $V_d(\boldsymbol{x}(k)) = \boldsymbol{x}(k)^T \hat{\boldsymbol{D}}\, \boldsymbol{x}(k)$ with $\hat{\boldsymbol{D}} = \mathbf{diag}\,[\hat{\boldsymbol{d}}]$ and $\hat{\boldsymbol{d}} \in \mathbb{R}^n_+$. Because $\boldsymbol{x}(k+1)^T \hat{\boldsymbol{D}}\, \boldsymbol{x}(k+1) = \hat{\boldsymbol{d}}^T \mathbf{diag}\,[\boldsymbol{x}(k+1)]\boldsymbol{x}(k+1)$ and $\boldsymbol{x}(k)^T \hat{\boldsymbol{D}}\, \boldsymbol{x}(k) = \hat{\boldsymbol{d}}^T \mathbf{diag}\,[\boldsymbol{x}(k)]\boldsymbol{x}(k)$, the difference of $V_d(\boldsymbol{x}(k))$ can be expressed as

$$\begin{aligned} \Delta V_d(\boldsymbol{x}(k)) &= \boldsymbol{x}(k+1)^T \tilde{\boldsymbol{D}}\, \boldsymbol{x}(k+1) - \boldsymbol{x}(k)^T \tilde{\boldsymbol{D}}\, \boldsymbol{x}(k) \\ &= \hat{\boldsymbol{d}}^T \left(\mathbf{diag}\,[\boldsymbol{x}(k+1)]\boldsymbol{x}(k+1) - \mathbf{diag}\,[\boldsymbol{x}(k)]\boldsymbol{x}(k)\right) \\ &= \hat{\boldsymbol{d}}^T \boldsymbol{v}(k) \leq 0, \end{aligned} \tag{2.22}$$

with

$$\begin{aligned} \boldsymbol{v}(k) &= \boldsymbol{x}(k+1) \odot \boldsymbol{x}(k+1) - \boldsymbol{x}(k) \odot \boldsymbol{x}(k) \\ &= \mathbf{diag}\,[\boldsymbol{x}(k+1)]\boldsymbol{x}(k+1) - \mathbf{diag}\,[\boldsymbol{x}(k)]\boldsymbol{x}(k)\,. \end{aligned} \tag{2.23}$$

Equation (2.22) shows that the inner product of the vector $\boldsymbol{v}(k)$ with a vector $\hat{\boldsymbol{d}}$, $\hat{\boldsymbol{d}} \in \mathbb{R}^n_+$, is always less than zero. Therefore, all the transformed vectors $\boldsymbol{v}(k)$ are located within one negative halfspace $h^-_{\hat{d}}$ whose normal is $\hat{\boldsymbol{d}}$, indicating that the convex conic hull $\mathbf{cone}\,\mathcal{V} \subset h^-_{\hat{d}}$. This completes the proof to the theorem. ■

Lemma 2.4.1 shows that if there exists a DQLF for system (2.14) that is transformed from (2.1) with an orthogonal matrix, the system (2.1) also owns a QLF. In accordance with this fact, theorem 2.4.2 can be extended to the sufficient and necessary condition for the existence of a QLF, which is the theoretical fundamental of the proposed data-driven stability criterion and stated in the following theorem.

Theorem 2.4.3 *Consider the nonlinear discrete-time system (2.1) with an equilibrium point at $\boldsymbol{x}_e = \mathbf{0}$, $\boldsymbol{x}_e \in \Omega$. For every vector $\boldsymbol{x}(k) \in \Omega$, a new vector $\tilde{\boldsymbol{v}}(k)$ can be generated using the following calculation*

$$\tilde{\boldsymbol{v}}(k) = \tilde{\boldsymbol{x}}(k+1) \odot \tilde{\boldsymbol{x}}(k+1) - \tilde{\boldsymbol{x}}(k) \odot \tilde{\boldsymbol{x}}(k)\,, \tag{2.24}$$

where $\tilde{\boldsymbol{x}}(k) = \boldsymbol{\Phi}\boldsymbol{x}(k)$ and $\boldsymbol{\Phi}$ is an orthogonal matrix. Let $\tilde{\mathcal{V}}$ represent the complete vector set of $\tilde{\boldsymbol{v}}(k)$, $k = 1\,, \ldots, \infty$, and the symbol $\mathbf{cone}\,\tilde{\mathcal{V}}$ represent the convex conic hull (the smallest convex cone) for $\tilde{\mathcal{V}}$. There exists a QLF for system (2.1) within the domain Ω, if and only if there exists at least one orthogonal matrix $\boldsymbol{\Phi}$ such that $\mathbf{cone}\,\tilde{\mathcal{V}}$ is located in a negative halfspace h^- of $\mathbb{R}^n$.

Theorem 2.4.3 does not require explicitly an analytical form of the nonlinear function $\boldsymbol{f}(\cdot)$ in system (2.1), but the complete time histories of system states. This fact makes it possible to apply the above theorem in the data-driven context to judge quadratic stability.

2.4.3 Interpretation of the stability condition using polyhedral cones

The necessary and sufficient quadratic stability condition, which discusses the QLF existence within the considered region Ω and is introduced in the theorem 2.4.3, can be interpreted by use of the geometrical relationship between two polyhedral cones. This fact is discussed in detail in this section.

According to theorem 2.4.3, the concerned system is quadratic stable if and only if the convex conic hull $\mathbf{cone}\,\tilde{\mathcal{V}}$ is located in a negative halfspace. In detail, the quadratic stability requires the existence of a suitable orthogonal matrix $\boldsymbol{\Phi}$ and a vector $\boldsymbol{d} \in \mathbb{R}^n_+$ so that for all the transformed vectors in the data set $\tilde{\mathcal{V}}$, the following condition holds

$$\langle \boldsymbol{d},\, \tilde{\boldsymbol{v}}(k) \rangle \leq 0,\ \boldsymbol{d} \in \mathbb{R}^n_+,\ \tilde{\boldsymbol{v}}(k) \in \tilde{\mathcal{V}}. \tag{2.25}$$

The data set $\tilde{\mathcal{V}}$ is obtained by transforming *all* the states vectors $x(k)$ within Ω, as introduced in the beginning of subsection 2.4.2. At the time instant $t = r$, the region covered by the system states $\boldsymbol{x}(k)$ is only a subset of the region Ω, where the nonlinear mapping $\boldsymbol{f}(\cdot)$ is defined. Correspondingly, the set of the transformed vectors $\tilde{\boldsymbol{v}}(k)$ at the time instant $t = r$ is only a subset of the $\tilde{\mathcal{V}}$, which is denoted as $\tilde{\mathcal{V}}_{r-1}$, with $\tilde{\mathcal{V}}_{r-1} = \{\tilde{\boldsymbol{v}}(k)\}$, $k = 1, \ldots, r-1$.

It should be noted that both $\tilde{\mathcal{V}}_{r-1}$ and $\tilde{\mathcal{V}}$ have infinite elements, because $\tilde{\mathcal{V}}_{r-1}$ contains the data of all the system trajectories located in Ω at the time instant $t = r$, and there can be infinite trajectories within the region Ω. Hence these two data sets are identical if $r \to \infty$, i.e.,

$$\tilde{\mathcal{V}}_{r-1} = \tilde{\mathcal{V}},\ \text{if}\ r \to \infty. \tag{2.26}$$

Therefore, the inequality in (2.25) can be rewritten into the following form

$$\langle \boldsymbol{d},\, \tilde{\boldsymbol{v}}(k) \rangle \leq 0,\ \boldsymbol{d} \in \mathbb{R}^n_+,\ \tilde{\boldsymbol{v}}(k) \in \tilde{\mathcal{V}_{r-1}},\ k = 1, \ldots r-1,\ \text{and}\, r \to \infty. \tag{2.27}$$

On the other hand, the polar cone of $\mathbf{cone}\,\tilde{\mathcal{V}}_{r-1}$, denoted as $\mathbf{cone}\,\tilde{\mathcal{V}}^o_{r-1}$, can be represented as

$$\mathbf{cone}\,\tilde{\mathcal{V}}^o_{r-1} = \left\{ \boldsymbol{y} | \tilde{\boldsymbol{v}}^T \boldsymbol{y} \leq 0, \tilde{\boldsymbol{v}} \in \mathbf{cone}\,\tilde{\mathcal{V}}_{r-1},\ \boldsymbol{y} \in \mathbb{R}^n \right\}. \tag{2.28}$$

By comparing (2.28) with (2.27), it can be found that the inequalities in (2.27) are identical to the definition of $\mathbf{cone}\,\tilde{\mathcal{V}}^o_{r-1}$ except that $\boldsymbol{d}$ is defined within $\mathbb{R}^n_+$, while $\boldsymbol{y} \in \mathbb{R}^n$. In fact, because $\boldsymbol{d}$ is an arbitrary vector located in $\mathbb{R}^n_+$ and $\mathbb{R}^n_+$ is also a polyhedral cone, the geometrical meaning of (2.27) can be interpreted as the intersection of two polyhedral cones: one is the polar cone $\mathbf{cone}\,\tilde{\mathcal{V}}^o_{r-1}$; the other is the positive real space $\mathbb{R}^n_+$. Based on this fact, the quadratic stability condition can be given for the data-driven context into theorem 2.4.4.

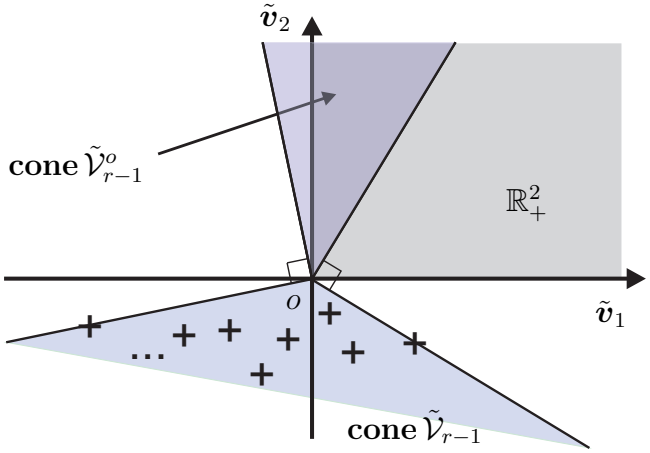


Figure 2.4: A negative halfspace in $\mathbb{R}^2$

Theorem 2.4.4 *The nonlinear discrete-time system (2.1) is quadratic stable if and only if there exists an orthogonal matrix* $\boldsymbol{\Phi}$ *such that at every time instant* $t = r$, $r \to \infty$, *the polar cone* $\mathbf{cone}\,\tilde{\mathcal{V}}^o_{r-1}$ *of the polyhedral cone* $\mathbf{cone}\,\tilde{\mathcal{V}}_{r-1}$ *constructed by the matrix* $\boldsymbol{\Phi}$ *and the system states* $\boldsymbol{x}(k)$ *at time instant* $t = r$, $\boldsymbol{x}(k) \in \Omega$, $k = 1, \ldots, r$, *follows the relationship*

$$\mathbf{cone}\,\tilde{\mathcal{V}}^o_{r-1} \cap \mathbb{R}^n_+ \neq \varnothing\,. \tag{2.29}$$

The real positive space $\mathbb{R}^2_+$, the polyhedral cone $\mathbf{cone}\,\tilde{\mathcal{V}}_{r-1}$ and its polar cone $\mathbf{cone}\,\tilde{\mathcal{V}}^o_{r-1}$ in the two-dimensional space are demonstrated in figure 2.4. Theorem 2.4.4 states that if the intersection between the set $\mathbf{cone}\,\tilde{\mathcal{V}}^o_{r-1}$ and $\mathbb{R}^n_+$ is not empty at every time instant, the system is quadratic stable. Furthermore, letting $\tilde{\boldsymbol{d}}$ be any vector located within $\mathbf{cone}\,\tilde{\mathcal{V}}^o_{r-1} \cap \mathbb{R}^n_+$, $r = \infty$, the QLF for this system can be expressed as

$$V(\boldsymbol{x}) = \boldsymbol{x}^{\boldsymbol{T}} \boldsymbol{\Phi}\, \mathbf{diag}\,[\tilde{\boldsymbol{d}}]\, \boldsymbol{\Phi}^T \boldsymbol{x}\,. \tag{2.30}$$

The condition (2.29) must be satisfied at every time instant for a quadratic stable system. Thus, to give a correct stability judgment to the considered motion, this criterion has to be examined at every time instant. If at some time instant this condition is not satisfied, the observed motion is judged as not quadratic stable.

It should be pointed out that the initial time of measurements is irrelevant to the final result of the stability judgement. By examining (2.28), it can be seen that $\mathbf{cone}\,\tilde{\mathcal{V}}^o_{r-1}$ is a convex cone by adding an inequality constraints $-\tilde{\boldsymbol{v}}(r-1)^T\,\boldsymbol{d} > 0$ to $\mathbf{cone}\,\tilde{\mathcal{V}}^o_{r-2}$, which implies that

$$\mathbf{cone}\,\tilde{\mathcal{V}}^o_{r-1} = \bigcap_{l=1}^{r-1} \mathbf{cone}\,\tilde{\mathcal{V}}^o_l\,. \tag{2.31}$$

Equation (2.31) shows that if $\mathbf{cone}\,\tilde{\mathcal{V}}_{r-1}^{o} \cap \mathbb{R}_{+}^{n} \neq \varnothing$ at $t = r$, the intersections between $\mathbb{R}_{+}^{n}$ and any of the cones $\mathbf{cone}\,\tilde{\mathcal{V}}_{l}^{o}$ formulated at former time instants $1 \leq l < r-1$, are inherently nonempty, which proves the irrelevance of initial judging time to the final result.

Theorem 2.4.4 indicates that the task of quadratic stability judgment is identical to determine whether there exists an orthogonal matrix $\mathbf{\Phi}$ so that the stability condition (2.29) can be satisfied at every time instant. At one certain time instant, if such a matrix exists, the concerned system can be judged as quadratic stable at the present time; and vice versa. In the next section, the stability condition is utilized to develop an online stability judgment algorithm used for the expert knowledge in the cognition-oriented stabilization.

2.5 Algorithm for online implementation

It should be mentioned that the stability condition stated in theorem 2.4.4 is only a theoretical condition and can hardly be examined in the practical sense. The reason is due to the following two conditions:

1) the data set $\tilde{\mathcal{V}}_{r-1}$ contains all the system trajectories within Ω at $t = r$ and therefore it has infinite elements; and

2) the system can be judged as stable in the sense of Lyapunov if and only if the time can reach infinity, i.e., $t = \infty$.

In the practical case of online stability judgment, both of these two conditions cannot be satisfied, because the data set of the measured system states, i.e., the data set $\mathcal{X}_r$ in the problem definition in section 2.2, contains only one system trajectory within the considered region Ω and the requirement of infinite time can never be realized in a practical sense. Therefore, if the region Ω in the theorem 2.4.4 is changed into the measured data set $\mathcal{X}_r$, the results of judgment can only be a necessary condition.

Despite of these facts, this section utilizes the data set $\mathcal{X}_r$ instead of Ω to construct the polyhedral cones $\mathbf{cone}\,\tilde{\mathcal{V}}_{r-1}^{o}$ and $\mathbf{cone}\,\tilde{\mathcal{V}}_{r-1}$ in the theorem 2.4.4, based on which the algorithm of stability judgment is developed, because the algorithm developed in this way can still be used as expert knowledge in the cognition-oriented stabilization. The detailed reasons and limitations for doing this are discussed in the next section.

2.5.1 Examining emptiness of the intersection between two cones

Suppose at first that an orthogonal matrix $\mathbf{\Phi}$ be available. As stated before, the original data set $\mathcal{X}_r$ can be transformed with $\mathbf{\Phi}$ into a new set $\tilde{\mathcal{V}}_{r-1}$ and the corresponding polar cone $\mathbf{cone}\,\tilde{\mathcal{V}}_{r-1}^{o}$ can be determined. The objective now is to design an algorithm to examine whether the condition (2.29) can be satisfied at different time instant.

Representation of the intersection

The V-representation of the polyhedral cone $\mathbf{cone}\,\tilde{\mathcal{V}}_{r-1}$ can be written as

$$\mathbf{cone}\,\tilde{\mathcal{V}}_{r-1} = \left\{ \sum_{k=1}^{r-1} \theta_k \tilde{\boldsymbol{v}}(k) | \tilde{\boldsymbol{v}}(k) \in \tilde{\mathcal{V}}_{r-1},\, \theta_k \geq 0,\, k = 1,\, \ldots,\, r-1 \right\}, \tag{2.32}$$

where θ_k is a non-negative number. According to the computational geometry theory [BV04], the H-representation of the polar cone $\mathbf{cone}\,\tilde{\mathcal{V}}^o_{r-1}$ can be established by using the matrix $\tilde{\boldsymbol{V}}_{r-1} \in \mathbb{R}^{(r-1)\times n}$ whose row vectors are the inverse of the elements in $\tilde{\mathcal{V}}_{r-1}$, represented as

$$\tilde{\boldsymbol{V}}_{r-1} = [-\tilde{\boldsymbol{v}}(k)]^T, \tag{2.33}$$

where $k = 1, \ldots (r-1)$ and $\tilde{\boldsymbol{v}}(k)$ are the elements in the vector set $\tilde{\mathcal{V}}_{r-1}$. Correspondingly, the form of the polar cone $\mathbf{cone}\,\tilde{\mathcal{V}}^o_{r-1}$ given in equation (2.28) can be reformulated in the form of matrix inequalities, as

$$\mathbf{cone}\,\tilde{\mathcal{V}}^o_{r-1} = \mathbf{cone}(\tilde{\boldsymbol{V}}_{r-1}) = \left\{ \boldsymbol{y} \,|\, \tilde{\boldsymbol{V}}_{r-1}\, \boldsymbol{y} \geq \boldsymbol{0},\ \boldsymbol{y} \in \mathbb{R}^n \right\}. \tag{2.34}$$

As mentioned in the section introducing geometric preliminaries, the positive real space $\mathbb{R}^n_+$ in (2.29) can be represented by using the unity matrix $\boldsymbol{I} \in \mathbb{R}^{n\times n}$ in the form of a polyhedral cone, as

$$\mathbb{R}^n_+ = \mathbf{cone}(\boldsymbol{I}) = \{ \boldsymbol{y} \,|\, \boldsymbol{I}\, \boldsymbol{y} \geq \boldsymbol{0},\ \boldsymbol{y} \in \mathbb{R}^n \}. \tag{2.35}$$

Because both $\mathbf{cone}\,\tilde{\mathcal{V}}^o_{r-1}$ and $\mathbb{R}^n_+$ are polyhedral cones, their intersection is also a polyhedral cone, which can be represented by taking advantage of the inequality form of $\mathbf{cone}\,\tilde{\mathcal{V}}^o_{r-1}$ and $\mathbb{R}^n_+$ defined in (2.34) and (2.35), respectively. Construct a new matrix $\boldsymbol{B}_{r-1} \in \mathbb{R}^{(n+p)\times n}$ by concatenate the matrix $\boldsymbol{V}_{r-1}$ defined in equation (2.33) and the identity matrix $\boldsymbol{I}$, as

$$\boldsymbol{B}_{r-1} = \begin{bmatrix} \tilde{\boldsymbol{V}}_{r-1} \\ \boldsymbol{I} \end{bmatrix}. \tag{2.36}$$

The intersection set between $\mathbb{R}^n_+$ and $\mathbf{cone}\,\tilde{\mathcal{V}}^o_{r-1}$ can be represented as the polyhedral cone determined by matrix $\boldsymbol{B}_{r-1}$, as

$$\mathbf{cone}(\boldsymbol{B}_{r-1}) = \{ \boldsymbol{y} \,|\, \boldsymbol{B}_{r-1}\boldsymbol{y} \geq \boldsymbol{0},\ \boldsymbol{y} \in \mathbb{R}^n \}. \tag{2.37}$$

Thus, according to theorem 2.4.4 it can be seen that the system is quadratic stable if and only if the polyhedral cone $\mathbf{cone}(\boldsymbol{B}_{r-1})$ is not empty, i.e., $\mathbf{cone}(\boldsymbol{B}_{r-1}) \neq \varnothing$, for $r = 1, \ldots, \infty$.

Determining the emptiness of the intersection

According to the computational geometry theory, to determine the emptiness of a polyhedral cone can be dealt with according to the Gordan's Theorem [Sto73], which is a collary of the famous Farkas' Lemma [Zie97] and stated as follows.

Theorem 2.5.1 (Gordan's Theorem) *Let $\boldsymbol{A}$ be an $m \times n$ matrix, $\boldsymbol{x}$ an n-dimensional vector, and $\boldsymbol{y}$ an m-dimensional vector. Either $\boldsymbol{Ax} > \boldsymbol{0}$ has a solution, or $\boldsymbol{Ay} = \boldsymbol{0}$, $\boldsymbol{y} \geq \boldsymbol{0}$ has a solution, but never both.*

Define a quadratic function $\varphi(\boldsymbol{\eta}) = \boldsymbol{\eta}^T \boldsymbol{A}\boldsymbol{A}^T\boldsymbol{\eta} = \boldsymbol{\eta}^T \boldsymbol{R}\boldsymbol{\eta}$, where $\boldsymbol{R} = \boldsymbol{A}\boldsymbol{A}^T$ and $\boldsymbol{\eta}$ is an m-dimensional vector. Clearly the value of $\varphi(\boldsymbol{\eta})$ is non-negative and the matrix $\boldsymbol{R}$ is a symmetric and positive definite matrix. It is shown in [Sto73] that the emptiness of the polyhedral cone with the inequality set $\boldsymbol{Ax} > \boldsymbol{0}$ as its interior can be determined by solving the quadratic programming problem

$$\begin{array}{ll} \min\limits_{\boldsymbol{\eta}} . & \varphi(\boldsymbol{\eta}) = \boldsymbol{\eta}^T \boldsymbol{R}\, \boldsymbol{\eta} \\ s.t. & \sum_{i=1}^{m} \eta_i = 1, \text{ and}, \\ & \eta_i \geq 0,\ i = 1, 2, ...m\,, \end{array} \tag{2.38}$$

where η_i represents the components of the vector $\boldsymbol{\eta}$. Let $\boldsymbol{\eta}^*$ be a solution to the above problem. The polyhedral cone defined by the inequality set $\boldsymbol{Ax} \geq \boldsymbol{0}$ is non-empty if and only if $\varphi(\boldsymbol{\eta}^*) > 0$. Moreover, the vector $\boldsymbol{A}^T\boldsymbol{\eta}^*$ is located in the interior of the regarded polyhedral cone.

The reason for this fact can be understood in the following way: If the inequality set $\boldsymbol{Ax} > \boldsymbol{0}$ mentioned in the theorem 2.5.1 has a solution, the value of $\phi(\boldsymbol{\eta})$ must be greater than zero, which indicates that the interior of the polyhedral cone defined by the inequality set $\boldsymbol{Ax} > \boldsymbol{0}$ is not empty. On the other hand, if the value of $\phi(\boldsymbol{\eta})$ is equal to zero, because $\boldsymbol{R}$ is positive definite, it can be concluded that there must exist a solution to the equation $\boldsymbol{A}^T\boldsymbol{\eta} = \boldsymbol{0}$. In this case, based on the theorem 2.5.1 it can be concluded that the inequality set $\boldsymbol{Ax} > \boldsymbol{0}$ has no solution, which implies that the polyhedral cone defined by the inequality set $\boldsymbol{Ax} > \boldsymbol{0}$ has an empty interior.

As far as this contribution is concerned, the emptiness of the polyhedral cone $\mathbf{cone}(\boldsymbol{B}_{r-1})$ defined in equation (2.37) should be determined. Based on the discussion above, the corresponding quadratic programming problem with respect to $\mathbf{cone}(\boldsymbol{B}_{r-1})$ should be detailed as the following optimization problem

$$\begin{array}{ll} \min\limits_{\boldsymbol{\alpha}} . & \varphi(\boldsymbol{\alpha}) = \boldsymbol{\alpha}^T \boldsymbol{M}\, \boldsymbol{\alpha} \\ s.t. & \sum_{i=1}^{n+r-1} \alpha_i = 1, \text{ and}, \\ & \alpha_i \geq 0,\ i = 1, 2, \ldots, n + r - 1\,, \end{array} \tag{2.39}$$

where $\boldsymbol{M}$ is determined by $\boldsymbol{M} = \boldsymbol{B}_{r-1}\boldsymbol{B}_{r-1}^T$ and $\boldsymbol{\alpha} \in \mathbb{R}^{n+r-1}$ is the vector of optimization variables. Denoting the solution to the problem (2.39) as $\boldsymbol{\alpha}^*$, from former discussion it can be seen that the polyhedral cone $\mathbf{cone}(\boldsymbol{B}_{r-1})$ is not empty if and only if the optimized value of $\varphi(\boldsymbol{\alpha}^*)$, denoted as φ^*, is greater than zero.

Furthermore, denoting the solution to the optimization problem (2.39) as $\boldsymbol{\alpha}^*$ and the vector $\boldsymbol{B}_{r-1}^T\boldsymbol{\alpha}^*$ as y^*, if $\varphi(\boldsymbol{\alpha}^*) > 0$, the vector $\boldsymbol{y}^*$ is located inside the cone $\mathbf{cone}(\boldsymbol{B}_{r-1})$. If for $r = 1, \ldots, \infty$, $\mathbf{cone}(\boldsymbol{B}_{r-1})$ shares an identical vector $\boldsymbol{y}^*$, then $\tilde{\boldsymbol{d}} = \boldsymbol{y}^*$ and a corresponding QLF can be given according to (2.30).

From the discussion in this subsection it can be concluded that the emptiness of the intersection between the polyhedral cone $\mathbf{cone}\,\tilde{\mathcal{V}}_{r-1}^o$ and the positive real space $\mathbb{R}_+^n$ can be determined by solving the quadratic programming problem (2.39). If the optimized value of the optimization problem (2.39) is greater than zero, the interior of the intersection is not empty and shows that there exists at least one QLF for the concerned motion of the system (2.1) at the time instant $t = r$. If the optimized value of the problem (2.39) is positive at every time instant, the concerned motion is globally quadratic stable; if at any time instant the optimized value is not positive, then the concerned motion is not quadratic stable.

2.5.2 Improving efficiency of numerical calculations

It is well known that the calculation burden of solving quadratic programming problems increases dramatically if the kernel matrix (i.e., the matrix given in the quadratic function, such as the matrix $\boldsymbol{R}$ in problem (2.38)) has a large dimension or is badly conditioned, which is critical to online implementations. Therefore, for the sake of calculation efficiency it is necessary to consider the proper way to construct the kernel matrix.

The efficiency of quadratic programming can be improved to avoid large-size and bad-conditioned kernel matrices. In the quadratic programming problem (2.39) concerned in this contribution, the kernel matrix is the matrix $\boldsymbol{M}$ defined $\boldsymbol{M} = \boldsymbol{B}_{r-1}\boldsymbol{B}_{r-1}^T$, where the matrix $\boldsymbol{B}_{r-1}$ is constructed from the vectors contained by the set $\tilde{\mathcal{V}}_{r-1}$, as shown in equation (2.36). Thus, in order to obtain a smaller-size and better-conditioned kernel matrix $\boldsymbol{M}$ for numerical calculation and improve the calculation efficiency, the matrix $\boldsymbol{B}_{r-1}$ should be constructed by choosing and manipulating the vectors in $\tilde{\mathcal{V}}_{r-1}$ in a proper manner, which is detailed in the following parts of this subsection.

Reducing the data amount

The upper part of matrix $\boldsymbol{B}_{r-1}$ is the matrix used in the H-representation of the polar cone $\mathbf{cone}\,\tilde{\mathcal{V}}_{r-1}^o$, as shown in equation (2.36). From equation (2.31) it can be seen

that the construction of the polyhedral cone $\mathbf{cone}\,\tilde{\mathcal{V}}^o_{r-1}$ requires the complete data set $\mathcal{X}_r$ containing the system state vectors at every time instant. While the size of the data set $\mathcal{X}_r$ is increasing with the time, the calculation burden of the construction of $\mathbf{cone}\,\tilde{\mathcal{V}}^o_{r-1}$ is also increasing because of the increasing data amount. Therefore, it is necessary to reduce the data amount for the construction of $\mathbf{cone}\,\tilde{\mathcal{V}}^o_{r-1}$, which can be realized according to the Carathéodory theorem [Car11].

Theorem 2.5.2 (Carathéodory Theorem) *Consider two proper convex cones* $\mathbf{cone}\,\mathcal{C}$ *and* $\mathbf{cone}\,\mathcal{C}'$*, where* $\mathcal{C}'$ *is a subset of* $\mathcal{C}$*. If a vector* $\boldsymbol{c} \in \mathbf{cone}\,\mathcal{C}$*, then* $\boldsymbol{c} \in \mathbf{cone}\,\mathcal{C}'$ *holds for the subset* $\mathcal{C}' \subseteq \mathcal{C}$ *of at most rank*

$$\mathbf{rank}\left(\begin{bmatrix} \mathbf{1} \\ \boldsymbol{C} \end{bmatrix}\right) = \mathbf{dim}\,(\mathbf{cone}\,\mathcal{C}) + 1 \tag{2.40}$$

vectors in $\boldsymbol{C}$*, where* $\boldsymbol{C}$ *is a matrix defined as* $\boldsymbol{C} = [\boldsymbol{c}_i]$ *and* $\mathbf{1}$ *represents the row vector whose entries are all equal to* 1 *and has the same dimension as the number of columns of the matrix* $\boldsymbol{C}$*.*

The Carathéodory theorem indicates that if the polyhedral cone defined by a convex set is proper and can satisfy the condition given in the Carathéodory theorem, it is identical to the polyhedral cone determined by the maximal linearly independent set of the newly constructed set $\tilde{\mathcal{V}}^c_{r-1}$, defined as

$$\begin{aligned} \tilde{\mathcal{V}}^c_{r-1} &= \left\{\tilde{\boldsymbol{v}}^c_1,\, \tilde{\boldsymbol{v}}^c_2,\, \ldots,\, \tilde{\boldsymbol{v}}^c_{r-1}\right\} \\ &= \left\{\begin{bmatrix} 1 \\ \tilde{\boldsymbol{v}}_1 \end{bmatrix}, \begin{bmatrix} 1 \\ \tilde{\boldsymbol{v}}_2 \end{bmatrix}, \ldots, \begin{bmatrix} 1 \\ \tilde{\boldsymbol{v}}_{r-1} \end{bmatrix}\right\}, \end{aligned} \tag{2.41}$$

where $\tilde{\boldsymbol{v}}^c_i,\ i = 1, \ldots, r-1$, represents the element of the data set $\tilde{\mathcal{V}}^c_{r-1}$ and is constructed by concatenating 1 above all the vectors $\tilde{\boldsymbol{v}} \in \tilde{\mathcal{V}}_{r-1}$.

Suppose that the maximal linearly independent set of $\tilde{\mathcal{V}}^c_{r-1}$ contain p linearly independent vectors, which is denoted as

$$\tilde{\mathcal{V}}^c_{r-1,\,\mathrm{in}} = \left\{\tilde{\boldsymbol{v}}^c_{1,\,\mathrm{in}},\, \tilde{\boldsymbol{v}}^c_{2,\,\mathrm{in}},\, \ldots,\, \tilde{\boldsymbol{v}}^c_{p,\,\mathrm{in}}\right\}, \tag{2.42}$$

where $\tilde{\mathcal{V}}^c_{r-1,\,\mathrm{in}}$ is used to denote the maximal linearly independent set of $\tilde{\mathcal{V}}^c_{r-1}$ and $\tilde{\boldsymbol{v}}^c_{i,\,\mathrm{in}},\ i = 1, \ldots, p$, represents the elements of the set $\tilde{\mathcal{V}}^c_{r-1,\,\mathrm{in}}$. Therefore, the polyhedral cone $\mathbf{cone}\,\tilde{\mathcal{V}}_{r-1}$ defined in equation (2.32) is identical to the cone defined as follows

$$\mathbf{cone}\,\tilde{\mathcal{V}}^c_{r-1,\,\mathrm{in}} = \left\{\sum_{i=1}^{p} \theta_i \tilde{\boldsymbol{v}}^c_{i,\,\mathrm{in}} \mid\ \tilde{\boldsymbol{v}}^c_{i,\,\mathrm{in}} \in \tilde{\mathcal{V}}^c_{r-1,\,\mathrm{in}},\, \theta_i \geq 0,\, i = 1, \ldots, p\right\}. \tag{2.43}$$

Correspondingly, the polar cone of $\mathbf{cone}\,\tilde{\mathcal{V}}_{r-1}$ is also identical to the polar cone of $\mathbf{cone}\,\tilde{\mathcal{V}}^c_{r-1,\,\mathrm{in}}$. Thus, instead of its former representation given in equation (2.34) with respect to the matrix $\tilde{\boldsymbol{V}}_{r-1}$, the H-representation of the polar cone $\mathbf{cone}\,\tilde{\mathcal{V}}^o_{r-1}$ can be established by using the matrix whose row elements are the same as the vector $\tilde{\boldsymbol{v}}$ used in the maximal linearly independent set $\tilde{\mathcal{V}}^c_{r-1,\,\mathrm{in}}$.

Taking those vectors $\tilde{\boldsymbol{v}}$ outside the maximal linearly independent set $\tilde{\mathcal{V}}^c_{r-1,\,\mathrm{in}}$ by removing the first element of $\tilde{\boldsymbol{v}}^c_{i,\,\mathrm{in}}$, $i = 1, \ldots, p$, a new vector set can be formulated as

$$\tilde{\mathcal{V}}_{r-1,\,\mathrm{in}} = \left\{\tilde{\boldsymbol{v}}_{1,\,\mathrm{in}},\, \tilde{\boldsymbol{v}}_{2,\,\mathrm{in}},\, \ldots,\, \tilde{\boldsymbol{v}}_{p,\,\mathrm{in}}\right\}, \tag{2.44}$$

where $\tilde{\boldsymbol{v}}_{i,\,\mathrm{in}}$, $i = 1, \ldots, p$, represents the elements of the set $\tilde{\mathcal{V}}_{r-1}$ used in $\tilde{\mathcal{V}}^c_{r-1,\,\mathrm{in}}$. It should be noted that the vectors $\tilde{\boldsymbol{v}}_{i,\,\mathrm{in}}$, $i = 1, \ldots, p$, are not necessary to be linearly independent and the set $\tilde{\mathcal{V}}_{r-1,\,\mathrm{in}}$ is not necessary to be the maximal linearly independent vector set of the set $\tilde{\mathcal{V}}_{r-1}$.

Define the matrix $\tilde{\boldsymbol{V}}_{r-1,\,\mathrm{in}} \in \mathbb{R}^{p\times n}$ as

$$\tilde{\boldsymbol{V}}_{r-1,\,\mathrm{in}} = [-\tilde{\boldsymbol{v}}_{i,\,\mathrm{in}}]^T, \tag{2.45}$$

where $\tilde{\boldsymbol{v}}_{i,\,\mathrm{in}}$, $i = 1, \ldots p$, is defined in equation (2.44). The polar cone $\mathbf{cone}\,\tilde{\mathcal{V}}^o_{r-1}$ can be reformulated with respected to $\tilde{\boldsymbol{V}}_{r-1,\,\mathrm{in}}$ as

$$\begin{aligned} \mathbf{cone}\,\tilde{\mathcal{V}}^o_{r-1} &= \mathbf{cone}(\tilde{\boldsymbol{V}}_{r-1,\,\mathrm{in}}) \\ &= \left\{\boldsymbol{y}\,|\,\tilde{\boldsymbol{V}}_{r-1,\,\mathrm{in}}\,\boldsymbol{y} \geq \boldsymbol{0},\ \boldsymbol{y} \in \mathbb{R}^n\right\}. \end{aligned} \tag{2.46}$$

Due to the fact that the size of $\tilde{\mathcal{V}}_{r-1,\,\mathrm{in}}$ is no larger than the size of $\tilde{\mathcal{V}}_{r-1}$, i.e., $p \leq r-1$, the data amount to construct the polar cone $\mathbf{cone}\,\tilde{\mathcal{V}}^o_{r-1}$ can be reduced by using the set $\tilde{\mathcal{V}}_{r-1,\,\mathrm{in}}$, instead of the original vector set $\tilde{\mathcal{V}}_{r-1}$.

In fact, the vectors of the set $\tilde{\mathcal{V}}_{r-1,\,\mathrm{in}}$ are the same as the extreme rays of the polyhedral cone $\mathbf{cone}\,\tilde{\mathcal{V}}_{r-1}$. Hence, this data amount reduction strategy is actually to take only the extreme rays of $\mathbf{cone}\,\tilde{\mathcal{V}}_{r-1}$ into mathematical calculation, neglecting all the vectors located inside $\mathbf{cone}\,\tilde{\mathcal{V}}_{r-1}$.

If the vector $\tilde{\boldsymbol{v}}(r)$ transformed from the system state vectors at the time instant $t = r+1$ is located within $\mathbf{cone}\,\tilde{\mathcal{V}}_{r-1,\,\mathrm{in}}$, it would be neglected and the old cone $\mathbf{cone}\,\tilde{\mathcal{V}}_{r-1,\,\mathrm{in}}$ would be used to judge the stability of the system motion at $t = r+1$. Only when the new vector $\tilde{\boldsymbol{v}}(r)$ is not located in the old polyhedral cone, it would then be considered and used to update the old polyhedral cone $\mathbf{cone}\,\tilde{\mathcal{V}}_{r-1,\,\mathrm{in}}$. Therefore, from the point of view of cognitive systems, the polyhedral cone $\mathbf{cone}\,\tilde{\mathcal{V}}_{r-1,\,\mathrm{in}}$ and its extreme rays can be treated as one kind of learned knowledge representing the stability of the experienced system motion.

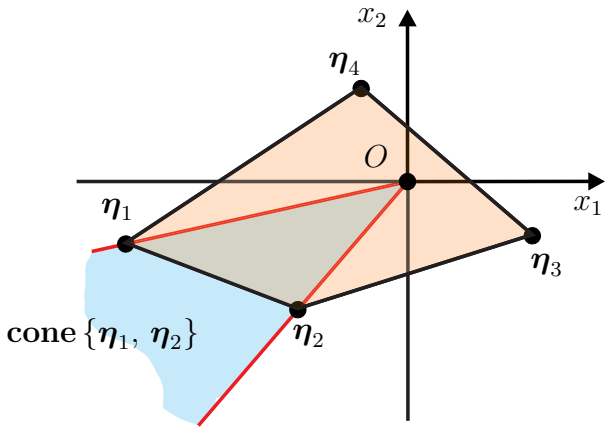


Figure 2.5: Example of unpointed cone

Checking properness before reducing the data amount

One of the condition in the Carathéodory theorem is that the considered convex cone should be pointed. Thus, it should be remarked that if **cone** $\tilde{\mathcal{V}}_{r-1}$ is not pointed, or the origin is located outside the polytope determined by $\tilde{\mathcal{V}}_{r-1}$, the cone **cone** $\tilde{\mathcal{V}}_{r-1}$ is not proper and the polyhedral cone determined by the vector set $\tilde{\mathcal{V}}_{r-1,\,\text{in}}$ is not identical to **cone** $\tilde{\mathcal{V}}_{r-1}$.

According to the polytope theory [Zie97], the cone $\tilde{\mathcal{V}}_{r-1}$ is pointed if there exists no such a vector which is denoted as $\tilde{\mathcal{V}}_{r-1}$ and satisfies the condition $-\tilde{\boldsymbol{v}}_p \in \tilde{\mathcal{V}}_{r-1}$. This condition can be examined by determining wether the origin is located in the convex hull (the polytope) constructed by the vectors of $\tilde{\mathcal{V}}_{r-1}$: the polyhedral cone is pointed if and only if the origin is outside the polytope. This condition can be verified by many methods proposed in computational geometry, such as the gift-wrappping algorithm [CLRS90], Fukuda algorithm [AF92], and so forth.

For example, suppose that the set $\tilde{\mathcal{V}}_{r-1}$ have four vectors: $\boldsymbol{\eta}_i$, $i = 1, \ldots, 4$, the positions of which are shown in figure 2.5. The yellow shaded area in figure 2.5 represents the polytope determined by this vector set. It can be immediately seen that the origin is located within this polytope, and the polytope, a subset of the convex cone determined by this $\tilde{\mathcal{V}}_{r-1}$, is not located in a negative halfspace, which implies the system is not quadratic stable.

As a consequence, it is necessary before the reduction to check that the polyhedral cone determined by the vector set $\tilde{\mathcal{V}}_{r-1}$ is pointed. If the polyhedral cone determined by $\tilde{\mathcal{V}}_{r-1}$ is not pointed, no reduction of data amount should be made, because in this case the cone $\tilde{\mathcal{V}}_{r-1}$ can never be located in a negative half space. As a result, if $\tilde{\mathcal{V}}_{r-1}$ is not pointed, the system motion can be judged directly as not quadratic stable and no further calculations are required.

Building well-conditioned kernel matrix

Vectors $\tilde{\boldsymbol{v}}_{i,\,\mathrm{in}}$, $i = 1, \ldots p$, whose inverse are the row vectors of the matrix $\tilde{\boldsymbol{V}}_{r-1,\,\mathrm{in}}$, are constructed from the measured system states following the equation (2.24). Because system states may vary from large values to very small values, the values of the elements of $\tilde{\boldsymbol{v}}_{i,\,\mathrm{in}}$ may also vary in a large range. More importantly, if the system states are near the origin in the state space, some values of $\tilde{\boldsymbol{v}}_{i,\,\mathrm{in}}$ may become extreme small, because the transformation defined in equation (2.24) contains power calculations. These facts may lead to the problem that with the system states approaching to the concerned equilibrium point (usually taken as the origin in the state space by shifting coordinates properly), some components in the kernel matrix $\boldsymbol{M}$ become almost zero, while some others are kept with relatively greater values.

This can weaken the positive definiteness of matrix $\boldsymbol{M}$ and increase its singularity by introducing new elements comparably small to zero, which can increase the difficulties to obtain accurate numerical solutions to the optimization problem (2.39) or even cause calculation failures. In other words, if system states are near zero, the kernel matrix may become ill-conditioned.

This problem can be solved by changing the representation of the polyhedral cone $\mathbf{cone}\,\tilde{\mathcal{V}}_{r-1,\,\mathrm{in}}$. Recalling the V-representation of $\mathbf{cone}\,\tilde{\mathcal{V}}_{r-1,\,\mathrm{in}}$ in equation (2.32), it can be seen that the scala θ_i, $i = 1, \ldots, p$, can be chosen arbitrary from the interval $[0,\ +\infty]$. This means the polyhedral cone $\mathbf{cone}\,\tilde{\mathcal{V}}_{r-1,\,\mathrm{in}}$ is only dependent on the directions of the vectors $\tilde{\boldsymbol{v}}_{i,\,\mathrm{in}}$, rather than their amplitudes.

Therefore, by defining the normalized vector of $\tilde{\boldsymbol{v}}_{i,\,\mathrm{in}}$, as

$$\bar{\boldsymbol{v}}_i = \nu \frac{\tilde{\boldsymbol{v}}_{i,\,\mathrm{in}}}{\left\|\tilde{\boldsymbol{v}}_{i,\,\mathrm{in}}\right\|_2}, \tag{2.47}$$

where ν is a positive scale and $\|\cdot\|_2$ represent the 2-nd norm calculation, the vector set $\tilde{\mathcal{V}}_{r-1,\,\mathrm{in}}$ can be normalized into the vector set $\bar{\mathcal{V}}_{r-1}$ whose elements are $\bar{\boldsymbol{v}}_i$. After normalization the values of the elements of $\bar{\boldsymbol{v}}_i$ are changed to comparable values with each other and can be shift from the neighborhood of the origin by using a larger value of ν.

Accordingly, the V-representation of $\mathbf{cone}\,\tilde{\mathcal{V}}_{r-1,\,\mathrm{in}}$ can be rewritten as follows

$$\begin{aligned} \mathbf{cone}\,\tilde{\mathcal{V}}_{r-1,\,\mathrm{in}} &= \mathbf{cone}\,\bar{\mathcal{V}}_{r-1} \\ &= \left\{ \sum_{i=1}^{p} \theta_i \bar{\boldsymbol{v}}_i \,|\, \bar{\boldsymbol{v}}_i \in \bar{\mathcal{V}}_{r-1},\, \theta_i \geq 0,\, i = 1, \ldots, p \right\}. \end{aligned} \tag{2.48}$$

Apparently, the cone $\mathbf{cone}\,\bar{\mathcal{V}}_{r-1}$ is identical to the cone $\mathbf{cone}\,\tilde{\mathcal{V}}_{r-1}$. Hence, the polar cone $\mathbf{cone}\,\tilde{\mathcal{V}}^o_{r-1}$ is identical to the polyhedral cone defined as

$$\mathbf{cone}(\bar{\boldsymbol{V}}_{r-1}) = \left\{ \boldsymbol{y} \,|\, \bar{\boldsymbol{V}}_{r-1}\, \boldsymbol{y} \geq \boldsymbol{0},\ \boldsymbol{y} \in \mathbb{R}^n \right\}, \tag{2.49}$$

where the matrix $\bar{\boldsymbol{V}}_{r-1}$ is defined as

$$\bar{\boldsymbol{V}}_{r-1} = [-\bar{\boldsymbol{v}}_i]^T . \tag{2.50}$$

If the polyhedral cone $\mathbf{cone}\,\tilde{\mathcal{V}}_{r-1}$ is pointed and the reduction of the data amount and the normalization of the concerned vector set have been made, it is then necessary to reformulate the kernel matrix $\boldsymbol{M}$ of the quadratic programming problems (2.39).

The matrix $\boldsymbol{B}_{r-1}$ defined in (2.36), which is used to represent the intersection and formulated the kernel matrix $\boldsymbol{M}$, can be rewritten by substituting the matrix $\tilde{\boldsymbol{V}}_{r-1}$ with $\bar{\boldsymbol{V}}_{r-1}$, i.e., the new matrix $\boldsymbol{B}_{r-1}$, denoted as $\bar{\boldsymbol{B}}_{r-1}$, can be constructed by the following equation

$$\bar{\boldsymbol{B}}_{r-1} = \begin{bmatrix} \bar{\boldsymbol{V}}_{r-1} \\ \boldsymbol{I} \end{bmatrix}. \tag{2.51}$$

The dimension of $\bar{\boldsymbol{B}}_{r-1}$ is $(n+p)\times n$. By substituting the matrix $\boldsymbol{B}_{r-1}$ with $\bar{\boldsymbol{B}}_{r-1}$ into the definition of $\boldsymbol{M}$, the new kernel matrix, denoted as $\bar{\boldsymbol{M}}$, can be simply calculated by

$$\bar{\boldsymbol{M}} = \bar{\boldsymbol{B}}_{r-1}\bar{\boldsymbol{B}}_{r-1}^T, \tag{2.52}$$

where $\bar{\boldsymbol{M}} \in \mathbb{R}^{(n+p)\times(n+p)}$. As a consequence, the quadratic programming problem (2.39) can be reformulated by replacing $\boldsymbol{M}$ with $\bar{\boldsymbol{M}}$, as

$$\begin{array}{ll} \min\limits_{\bar{\boldsymbol{\alpha}}}. & \bar{\varphi}(\bar{\boldsymbol{\alpha}}) = \bar{\boldsymbol{\alpha}}^T\,\bar{\boldsymbol{M}}\,\bar{\boldsymbol{\alpha}} \\ s.t. & \sum_{i=1}^{n+p}\alpha_i = 1, \text{ and}, \\ & \alpha_i \geq 0,\ i = 1, 2, \ldots, n+p, \end{array} \tag{2.53}$$

where $\bar{\boldsymbol{\alpha}} \in \mathbb{R}^{n+p}$ is the vector of optimization variable. The kernel matrix $\bar{\boldsymbol{M}}$ in (2.53) has smaller size and is better-conditioned than the old kernel matrix $\boldsymbol{M}$ in (2.39), by which way the calculation efficiency can be improved.

The above quadratic programming is identical to the optimization problem (2.39), which means that if the optimized value of $\bar{\varphi}$ in (2.53) with respect to its solution vector $\bar{\boldsymbol{\alpha}}^*$, denoted as $\bar{\varphi}^* = \bar{\varphi}(\bar{\boldsymbol{\alpha}}^*)$, is greater than zero, then the intersection between the positive real space $\mathbb{R}_+^n$ and the polar cone of $\mathbf{cone}\,\tilde{\boldsymbol{V}}_{r-1}$ is not empty and the vector $\bar{\boldsymbol{B}}^T\bar{\boldsymbol{\alpha}}^*$ is located in this intersection, and vice versa.

2.5.3 Introducing orthogonal constraints

Due to the fact that the kernel matrix $\bar{\boldsymbol{M}}$ in the optimization problem (2.53) depends on the choice of the orthogonal matrix $\boldsymbol{\Phi}$ contained in the transformation (2.24) in

theorem 2.4.3, as well as the matrix $\boldsymbol{M}$ in (2.39), the optimized value $\bar{\varphi}^*$ depends also on the orthogonal matrix $\boldsymbol{\Phi}$.

Therefore, from the discussion above it can be concluded that if the maximum of $\bar{\varphi}^*$ with respect to every $n \times n$ orthogonal matrix is not greater than zero, no suitable $\boldsymbol{\Phi}$ can be found to fulfil the stability condition (2.29), and vice versa. This fact indicates that when taking orthogonal constraints into consideration, the stability condition (2.29) can be examined by solving a max-min problem: to maximize the minimum of $\bar{\varphi}(\bar{\boldsymbol{\alpha}})$ with respect to the complete set of orthogonal matrices $\boldsymbol{\Phi}$ and the vector $\bar{\boldsymbol{\alpha}}$ with constraints given in the optimization problem (2.53). If the optimized value in this max-min problem has a positive sign at every time instant, the observed motion of the concerned system is quadratic stable, and vice versa.

The parametric representation of orthogonal matrixes proposed in [TL08] taken to construct $\boldsymbol{\Phi}$ in the max-min problem, because it is proven to be capable of covering *stochastically* the complete set of $n \times n$ orthogonal matrices. The construction process of this parametric orthogonal matrix is summarized in the appendix. This parametric representation of orthogonal matrices is composed of $n(n-1)/2$ parameters which can vary within the interval $[0,\ 2\pi)$ respectively. Use the $\boldsymbol{\zeta}$ to represent the vector composed of $\zeta_i \in [0,\ 2\pi)$, $i = 1,\ \ldots, n(n-1)/2$. By choosing each ζ_i from the normal distribution within $[0,\ 2\pi)$ randomly, the matrix $\boldsymbol{\Phi}$ constructed by this method is capable of representing every the $n \times n$ orthogonal matrix.

Thus, taking $\boldsymbol{\Phi}$ as a function of $\boldsymbol{\zeta}$, the max-min optimization can be formulated by introducing $\boldsymbol{\zeta}$ into (2.53) as

$$\begin{aligned} \max_{\boldsymbol{\zeta}} \min_{\bar{\boldsymbol{\alpha}}} \quad & \bar{\varphi}(\bar{\boldsymbol{\alpha}}\,, \boldsymbol{\zeta}) = \bar{\boldsymbol{\alpha}}^T\, \bar{\boldsymbol{M}}(\boldsymbol{\Phi}(\boldsymbol{\zeta}))\, \bar{\boldsymbol{\alpha}} \\ s.t. \quad & \zeta_i \in [0,\ 2\pi),\ i = 1,\ 2,\ \ldots,\ n(n-1)/2, \\ & \textstyle\sum_{j=1}^{n+p} \alpha_j = 1, \text{ and} \\ & \alpha_j \geq 0,\ j = 1,\ 2\,,\ \ldots,\ n+p\,. \end{aligned} \tag{2.54}$$

Because the optimization variable $\boldsymbol{\zeta}$ in (2.54) must be chosen stochastically, the gradient of the maximization problem cannot be obtained. Hence, the max-min problem developed above should correspondingly be solved with utilization of random optimization techniques like genetic algorithm and so forth.

Denoting the solution to the max-min optimization problem (2.53) as $\{\boldsymbol{\Phi}(\boldsymbol{\zeta}^*),\ \bar{\boldsymbol{\alpha}}^*\}$ and the optimized value as $\bar{\varphi}^{**}$, according to the discussion in the subsection 2.5.1, the convex cone $\mathbf{cone}(\mathbf{B_{r-1}})$ is not empty if and only if $\bar{\varphi}^{**} > 0$.

Based on the theorem 2.4.3, the concerned motion of system (2.1) is quadratic stable if and only if $\bar{\varphi}^{**} > 0$ at every time instant, which implies that the quadratic stability of the concerned motion of the unknown nonlinear system (2.1) can be determined by solving the max-min problem (2.53).

2.6 Supplementary remarks about the proposed criterion

2.6.1 Towards the assumption of full observability

In general, the assumption of full observability for the proposed data-driven quadratic stability criterion may be too strict to be fulfilled, because in reality it may be impossible to get full measurements of all the system states. However, if the input-output model of the system dynamics can represent the stability of the state space model,i.e., if the undetectable system states are inherently stable, the proposed criterion can also be used to judge the stability of the concerned motion, because an input-output model can be transformed into a state-space model and the stability judgment can be made with respect to the state vectors of the transformed state-space model.

The general input-output form of the discrete-time nonlinear system used in system identification can be expressed as

$$\boldsymbol{y}(k+1) = \boldsymbol{f}(\boldsymbol{y}(k),\, \boldsymbol{y}(k-1),\, \ldots,\, \boldsymbol{y}(k-n_y), \boldsymbol{u}(k),\, \boldsymbol{u}(k-1),\, \ldots,\, \boldsymbol{u}(k-n_u)), \quad (2.55)$$

where $\boldsymbol{u}$ represent the system input, $\boldsymbol{y}$ the output, $\boldsymbol{f}$ the nonlinear function, and n_u and n_y are the orders of input and output, respectively. Under the conditions when the system is Lipschutz and the partial derivative of the function $\boldsymbol{f}(\cdot)$ with respect to $\boldsymbol{u}$ and $\boldsymbol{y}$, the input-output form in (2.55) can be transformed into the state-space form whose state vector can be chosen from the input and output vectors. Some examples of the transformation from input-output data to state-space representation can be found in [FS01, CZH09].

The transformed state-space form can be described by the equation (2.1) concerned in this contribution. Although in the problem setting of data-driven stability judgment it is impossible to obtain the detailed state-space representation, as long as the structure of the state vectors is known, which can usually be satisfied before the transformation from input-output form to state-space representation, it is sufficient to use the proposed stability criterion to judge the system stability, because the stability judgment based on the input-output model is the same as that based on the transformed state-space representation.

2.6.2 Towards the necessity of judgment results

It has already been shortly mentioned in the beginning of section 2.5 that the judgment results of the proposed algorithm is only a necessary condition. The following parts of this subsection explains the reason for this necessity is explained in detail, as well as the possibility of utilizing this algorithm with its necessary stability judgments in control design and stability analysis.

Original definition of stability of motion

The original discussions of Lyapunov towards stability are developed upon the concept of *unperturbed motion* and *perturbed motion* [Lya92]. The discussion of stability of a motion is always in the sense of an unperturbed motion with a group of perturbations.

According to [Hah67], a motion can be represented as $\boldsymbol{x} = \boldsymbol{x}(t, \boldsymbol{\sigma}_a, t_0)$, where $\boldsymbol{\sigma}_a$, t_0, and t are parameters describing the motion. The parameter $\boldsymbol{\sigma}_a$ can range over some space to represent different motions. Holding $\boldsymbol{\sigma}_a$ fixed, the motion $\boldsymbol{x} = \boldsymbol{x}(t, \boldsymbol{,}t_0)$ is the family of *unperturbed motions*. If the parameter is allowed to range over a certain neighborhood of $\boldsymbol{\sigma}_a$, for instance a spherical neighorhood $\|\boldsymbol{\sigma}_b - \boldsymbol{\sigma}_a\| \leq r$, the motion $\boldsymbol{x}(t, \boldsymbol{\sigma}_b, t_0)$ is then called a *perturbed motion*. The unperturbed motion can be considered as the equilibrium of a nonlinear system. The unperturbed motion is *stable* (in the sense of Lyapunov), if for each $\varepsilon > 0$, there exists a $\delta > 0$ such that

$$d(t, t_0; \boldsymbol{\sigma}_a, \boldsymbol{\sigma}_b) < \delta, \tag{2.56}$$

provided only that $\|\boldsymbol{\sigma}_b - \boldsymbol{\sigma}_a\| \leq r$.

The following expression

$$d(t, t_0; \boldsymbol{\sigma}_a, \boldsymbol{\sigma}_b) := \|\boldsymbol{x}(t, \boldsymbol{\sigma}_b, t_0) - \boldsymbol{x}(t, \boldsymbol{\sigma}_a, t_0)\| \tag{2.57}$$

can be interpreted as the distance of the points on the two motions. Because by co-ordinates transformations the parameter $\boldsymbol{\sigma}_a$ can be changed to 0, the distance $d(t, t_0; \boldsymbol{\sigma}_a, \boldsymbol{\sigma}_b)$ can always be formulated as $d(t, t_0; \boldsymbol{\sigma}_a, \boldsymbol{\sigma}_b) = \boldsymbol{x}(t, \boldsymbol{\sigma}_b, t_0)$. As a result, the perturbed motion $\boldsymbol{x}(t, \boldsymbol{\sigma}_b, t_0)$ can be treated as the different solutions to the differential equation describing the system, with respect to the different parameter $\boldsymbol{\sigma}_b$ ranging in some neighborhood of $\boldsymbol{\sigma}_a$.

About the necessity and sufficiency in the proposed criterion

As pointed in [Hah67], the concept of stability has

> ... a definition very similar to that of the concept of continuous and we can actually think of stability as a type of continuity if we look at the motion as a whole and consider its dependence on $\boldsymbol{\sigma}_a$, resp. $\boldsymbol{\sigma}_b$.

This continuity-like characteristics of stability shows that a unperturbed motion (i.e., an equilibrium of a system) is called stable if *all* its perturbed motions with parameter $\boldsymbol{\sigma}_b$ ranging over a certain neighborhood of $\boldsymbol{\sigma}_a$ (ie., one certain neighborhood of the equilibrium) can approach to the unperturbed motion.

Because the proposed algorithm only considers the current running trajectory of a specific system (i.e., a specific perturbed motion with some fixed parameter $\boldsymbol{\sigma}_b$), which is only one trajectory within the neighborhood of the considered equilibrium (the region Ω in former discussions), this algorithm does not support the stability of the equilibrium in the sense of Lyapunov, which requires the parameter $\boldsymbol{\sigma}_b$ ranging over the whole neighborhood of $\boldsymbol{\sigma}_a$ (i.e., the complete region of Ω). From this point of view, the proposed criterion only a *necessary* condition with respect to the concerned equilibrium in the sense of Lyapunov.

Furthermore, the theorem 2.4.4 can be fulfilled if and only if times goes into infinity, which can not be examined in reality. If the judgment of the proposed algorithm at one certain time instant $t = r$ is unstable, it can be predicted that the judgment results for the cases $t > r$ are also unstable. However, if its judgment result at $t = r$ is stable, no prediction about stability for the case when $t > r$ can be made by the proposed algorithm. Therefore, the judgment of the proposed algorithm within finite time is also only a local judgement with respect to the time.

Practical utilization of the proposed algorithm

In spite of the necessity of its judgment results, the proposed algorithm has still practical sense in both online stability analysis and design of control.

In many practical problems of online stability assessment, the capital problem is to judge whether the perturbed motion of the concerned system can converges to the equilibrium. Taking the real-time stability assessment in power systems [Sav05] for example, the main task here is to judge whether the power system shall not shut down and can return to its original normal working conditions (the equilibrium) after a disturbance like the typical three-phase faults. In this case, there is only one motion being taken into consideration, which is exactly the case where the proposed stability criterion can be applied.

In the case of control, especially in the cognition-oriented stabilization proposed in this dissertation, the proposed algorithm can be used to design a controller which can stabilize the system to be controlled. If a controller satisfying the stability condition judged by the algorithm exists and can be found under different initial conditions within a neighborhood of the equilibrium, the controlled system shall possess a (both sufficiently and necessarily) stable equilibrium and thereby be stabilized.

From these perspectives it can be seen that despite of the necessity of the judgment results, the proposed algorithm can still be conditionally utilized in both online stability assessment and control design problems. In this contribution, this algorithm and the stability condition behind are utilized as the expert knowledge to search for a controller satisfying the stability condition determined by the proposed algorithm under different initial conditions, which will be introduced in the next chapter.

2.7 Numerical examples

2.7.1 Introducing examples of switched-linear system

Consider the switched-linear system given in [CGH03], which is constructed by three linear systems as

$$\dot{\boldsymbol{x}} = \begin{cases} \boldsymbol{A}_1\boldsymbol{x}\,, & \boldsymbol{x}_1^2 + \boldsymbol{x}_2^2 \leq 1 \\ \boldsymbol{A}_2\boldsymbol{x}\,, & 1 < \boldsymbol{x}_1^2 + 2\boldsymbol{x}_2^2 \leq 6 \\ \boldsymbol{A}_3\boldsymbol{x}\,, & \boldsymbol{x}_1^2 + \boldsymbol{x}_2^2 < 6 \end{cases}, \tag{2.58}$$

where the three linear matrices are defined as

$$\boldsymbol{A}_1 = \begin{bmatrix} -4 & -1 \\ 2 & 0.1 \end{bmatrix},$$
$$\boldsymbol{A}_2 = \begin{bmatrix} -2 & 3 \\ 1 & -2 \end{bmatrix}, \quad \text{and}$$
$$\boldsymbol{A}_3 = \begin{bmatrix} -5 & 2 \\ 3 & -2 \end{bmatrix}.$$

The observed data of one system trajectory with initial conditions $\boldsymbol{x}(0) = [-7\,,\ 8]^T$ are obtained by simulating the system for 8 seconds. The sampling time for the measurements is 0.1s. The data are assumed as noise free. In Fig. 2.6 the resulting trajectory is shown in the phase plane. The quadratic stability judgement at $t = 8$s can be obtained by taking this data set as input and solving the max-min problem (2.54). The max-min problem is solved by the genetic algorithm with binary encoding technique [Hol75]. The reason for choosing the genetic algorithm is because the orthogonal matrix $\boldsymbol{\Phi}$ is produced randomly, and the genetic algorithm is inherently a random optimization solver which is able to converge to the optimal solution with a probability approaching to 1. Because the system is of order two, there is only one parameter to be determined in the 2-dimensional orthogonal matrix $\boldsymbol{\Phi}$. This parameter, denoted as θ, is chosen randomly following a normal distribution within $[0,\ 2\pi)$.

The results show that when $\theta = 216^\circ$ and the corresponding orthogonal matrix equals to

$$\boldsymbol{\Phi} = \begin{bmatrix} -0.5878 & -0.8090 \\ -0.8090 & 0.5878 \end{bmatrix}, \tag{2.59}$$

the optimized value φ^{**} equals to $7.89 \times 10^{-2} > 0$, indicating the polyhedral cone is nonempty. There exists a common QLF for all these three linear systems. The data set $\mathcal{V}_{r-1}$ transformed from $\mathcal{X}_r$ with the above $\boldsymbol{\Phi}$ is shown in Fig. 2.7, from which

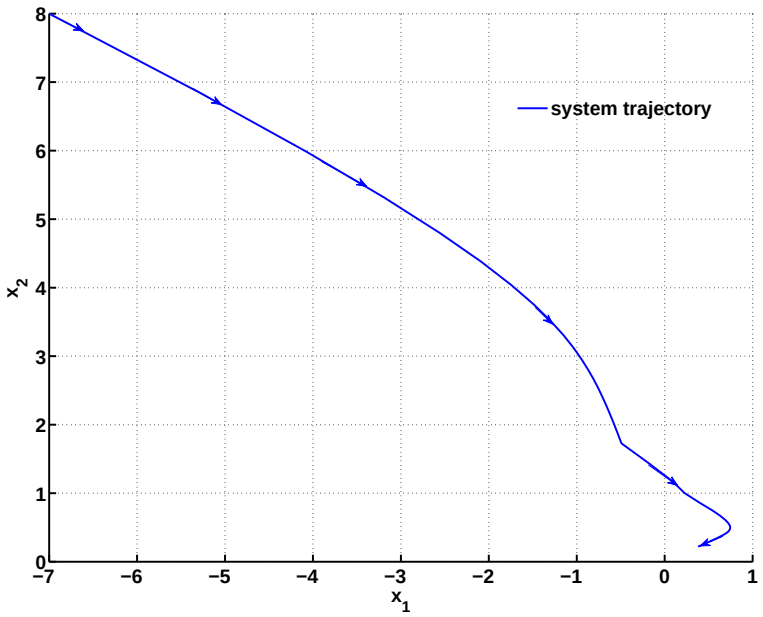


Figure 2.6: Observed trajectory of system (2.58)

it can be seen that the convex conic hull determined by the transformed vectors is located in a negative halfspace and concluded that at this time instant the system can be judged as quadratic stable.

Equation(2.30) can be used to calculate the matrix $\boldsymbol{P}$, as

$$\boldsymbol{P} = \begin{bmatrix} 0.0042 & 0.0034 \\ 0.0034 & 0.0064 \end{bmatrix} . \tag{2.60}$$

Although the function $V(\boldsymbol{x}) = \boldsymbol{x}^T \boldsymbol{P} \boldsymbol{x}$ constructed with the numerical value of $\boldsymbol{P}$ given in (2.60) can be proven analytically to be a QLF for each subsystem of (2.58) with Lyapunov stability theory, it must be pointed out that this QLF is just by coincidence correctly obtained. In fact, the QLF calculated by this way can only be correct if the optimization problem can be solved for $t \to \infty$. In other words, the QLF obtained by (2.58) at any single time instant $t \neq \infty$ is only correct for the current data set. Although this fact causes a strict limit of the proposed method to determine a correct QLF, it does not influence the results of the online stability judgement, which requires to be checked all the time. If at any time instant, the optimized value φ^{**} is identical to zero, the system will not be quadratic stable.

From this introducing example it is shown that the proposed stability criterion can be utilized in the online stability assessment of a two-dimensional nonlinear unknown system. It can also be seen from the discussion in the paragraph above that the

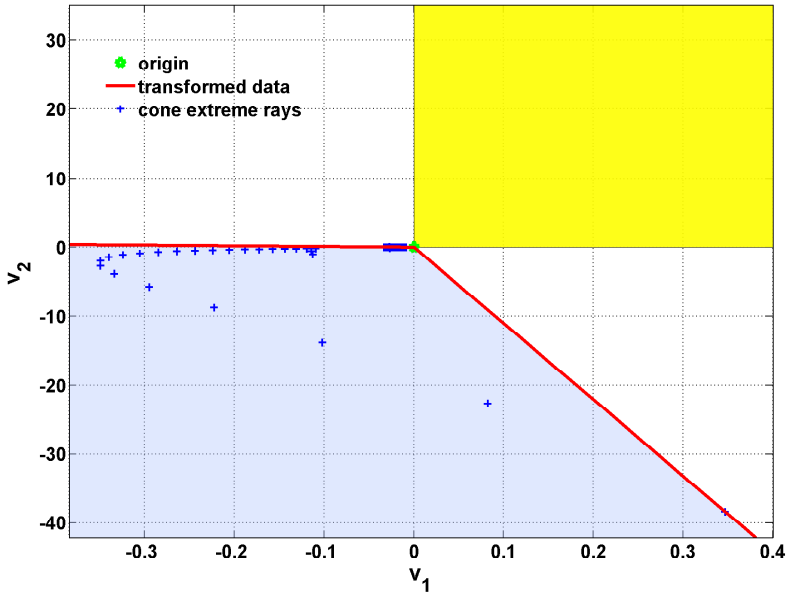


Figure 2.7: Convex hull constructed by observed data of system (2.58)

proposed method can also be used to find a quadratic stability candidate. In the next subsection, a three dimensional example is given to show the performance of the proposed method in higher-order systems.

2.7.2 Example of a nonlinear system with unstable limit cycle

The second example is a 3-dimensional nonlinear system [JW71]. The dynamical behavior of this system can be characterized by the existence of an unstable limit cycle oscillation. The mathematical description of this 3-dimensional system is expressed below as

$$\begin{aligned} \dot{x}_1 &= x_2\,, \\ \dot{x}_2 &= x_3\,, \\ \dot{x}_3 &= -x_3 - x_2 - 2x_1 - x_3 - x_2^2\,. \end{aligned} \tag{2.61}$$

The proposed method are executed to give data-driven stability judgements for two different trajectories of system (2.61) respectively. The initial condition of the first trajectory, $\boldsymbol{x}(0) = [-0.8\,,\ -0.4\,,\ 1.5]^T$, is inside the unstable limit cycle. Clearly this trajectory owns an stable equilibrium at the origin. This stable trajectory obtained at $t = 10$s is shown in the phase plane in Fig. 2.8. Using the data of

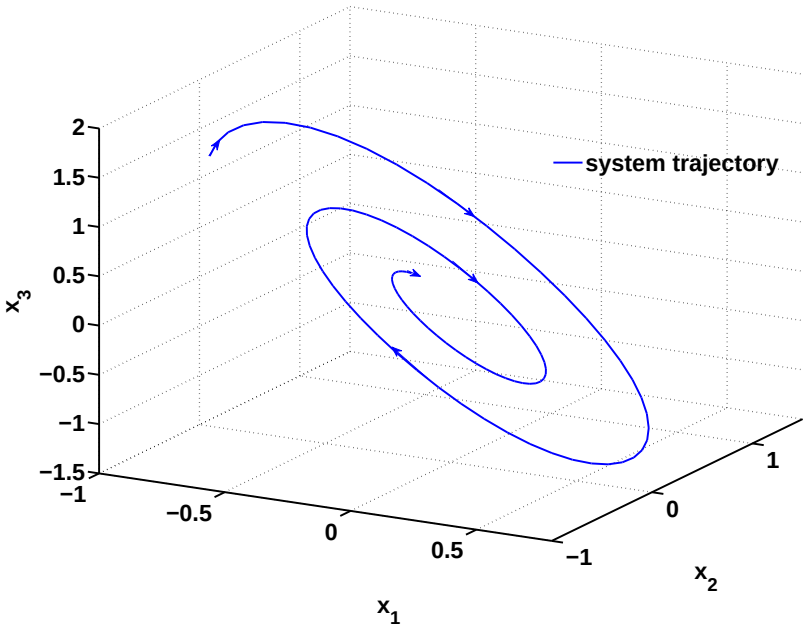


Figure 2.8: Observed trajectory of system (2.61) with initial point inside limit cycle

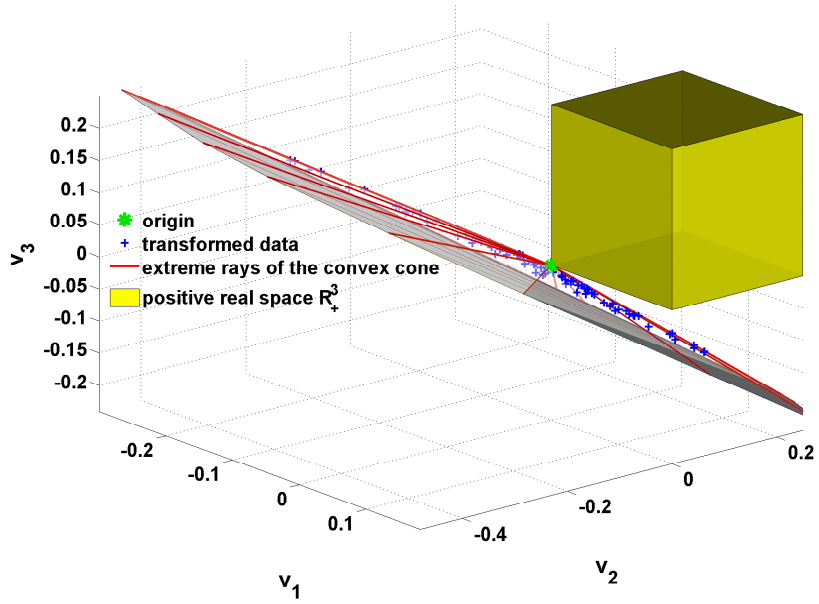


Figure 2.9: Convex hull constructed by the trajectory of system (2.61) with initial point inside limit cycle

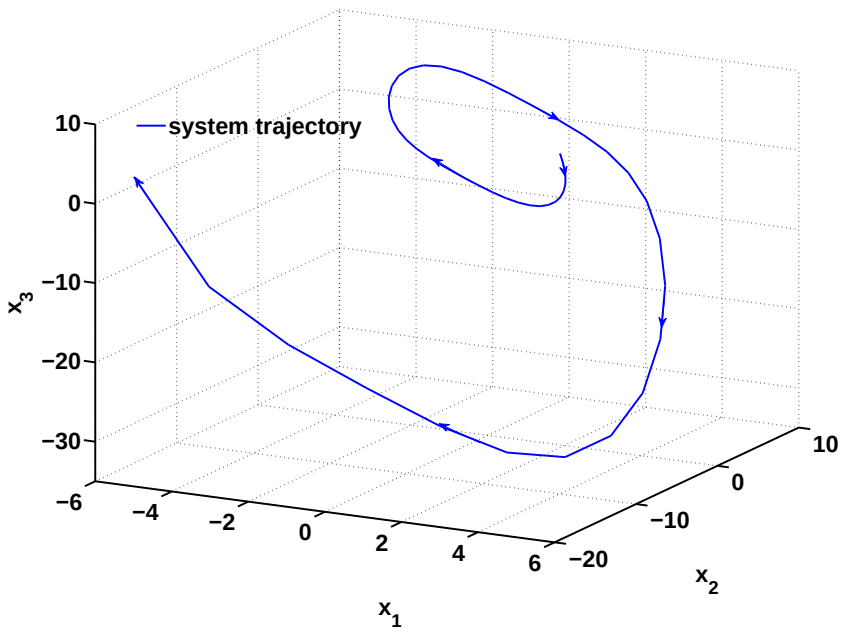


Figure 2.10: Observed trajectory of system (2.61) with initial point outside limit cycle

this trajectory, the proposed data-driven method is applied to judge the stability of the motion of system (2.61) with the corresponding initial condition at $t = 10$s. The optimization solver is the same as the one used in above subsection. The optimization results show that the optimized value φ^{**} can be obtained with using $\boldsymbol{\theta} = [37.5561^\circ,\ 0.6819^\circ,\ 4.8378^\circ]^T$ and is equal to 6.44×10^{-2}, $\varphi^{**} > 0$. The transformed vector set $\tilde{\mathcal{V}}_{r-1}$ and the corresponding convex conic hull are shown in Fig. 2.9.

It can be seen in Fig. 2.9 that the convex conic hull (smallest convex cone) determined by the transformed data-set $\tilde{\mathcal{V}}_{r-1}$ are located in a negative halfspace, which indicates that the observed motion at $t = 10$s can be judged as quadratic stable.

The second trajectory with the initial conditions $\boldsymbol{x}(0) = [1.8\,,\ 0.4\,,\ 1.5]^T$ is outside the limit cycle, which implies when time goes to infinity, the trajectory will also approach to infinity. This trajectory at $t = 5$s is shown in Fig. 2.10 and the corresponding data-set is given to the proposed method for stability judge. The optimized value φ^{**} equals to 0 for all the different values of $\boldsymbol{\theta}$, indicating this trajectory is not quadratic stable according to theorem 2.4.4. In fact, the origin is contained in the convex hull determined the transformed data-set, no matter which orthogonal matrix is used for the transformation. This means every hyperplane passing the origin will separate the transformed-data set into two parts. Thus, no negative halfspace containing the transformed data-set will exist and this motion is accordingly

classified as not quadratic stable.

The example of the three-dimensional system with unstable limit cycles shows that the proposed stability criterion concentrates on the stability with respect to certain equilibrium point, which is assumed as the origin in this thesis. It can be envisioned that when the proposed method is utilized in a system with stable limit cycles, all the trajectories shall be judged as unstable, because the origin is not a stable equilibrium to every the trajectory of the system. In other words, the online stability judgement method proposed in this thesis is limited to judge stability with respect to equilibriums, but not with respect to limit cycles.

2.8 Summary of this chapter

This chapter introduces the proposed data-driven quadratic stability criterion which can be used for online stability judgement. Based on the interpretation of quadratic stability that the existence of a QLF is identical to the existence of a suitable orthogonal matrix with which all the system states can be mapped into a negative halfspace, this thesis formulate this problem into an max-min problem which can identical to a geometrical problem of determining the emptiness of a generated polyhedral cone.

The detailed usage of this expert knowledge within the proposed cognition-oriented stabilization is introduced in the next chapter.

3 Realization of Cognition-Oriented Stabilization

It is mentioned in the introduction chapter that the realization of cognition-oriented stabilization rely on two aspects: expert knowledge about stability, and a cognitive architecture suitable for system control. As the expert knowledge adopted in this contribution has been introduced in last chapter, this chapter focuses on the cognitive architecture suitable for control and the mechanism how the proposed expert knowledge is embedded in this architecture to realize cognition-oriented stabilization.

3.1 The cognitive architecture suitable for control

3.1.1 Limitations of existing cognitive architecture

During the past two decades many cognitive architectures have been proposed based on different approaches and methodologies, such as Soar based on physical symbolic hypothesis [New90], ACT-R [ABB+04], EPIC [KWM97], and ICARUS [LC06] with the primary aim of producing artificial intelligence mimicking human cognition. These existing cognitive architectures can be used to describe a wide range of aspects of human cognition and many of them have also been put into engineering applications to guide the construction of cognitive systems.

Nevertheless, in spite of the prominent achievements of these cognitive studies and their successful applications, the existing cognitive architectures such as Soar and ACT-R do not support to the problem of stabilization of unknown systems, especially not related to nonlinear dynamical systems. The reason for this argument lie in one neglected aspect with respect to acquisition and utilization of knowledge in the existing cognitive architectures.

Most of the existing cognitive architectures utilize the incremental learning as the manner of knowledge acquisition. In the incremental learning case, the knowledge is updated after every new experience is obtained. However, the knowledge obtained in the incremental manner is usually local and sufficiently precise global knowledge can only be, if possible, obtained after a sufficient number of interaction cycles. Correspondingly, the utilization of this knowledge, such as based on it to generate deeper understanding of the external world or to plan actions, has also only a local sense before the precise global knowledge is learned.

If the external environment is unknown and dynamical, the precise global knowledge may be impossible to be obtained. Under such cases, it is necessary to develop a mechanism to organize the cognitive functions with respect to knowledge acquisition and utilization in a proper manner, so that the reactions produced by the architecture can converge to the desired situation. Here this mechanism is not dealing with

the detailed realization methods of cognitive functions, but the sequence of learning and planning functions within an interaction cycle and the corresponding hierarchy of the knowledge organization.

For example, the stabilization problem in control context is prominently sensitive to the order and hierarchy of learning functions. Here, the learned knowledge at one certain time instant contains the identified model of the plant dynamics and the controller's understanding about the stability of the plant motion, and the two corresponding learning functions are online system identification techniques and the stability judgement method. Suppose that the two learning functions are arranged in a sequential manner, i.e., the knowledge about stability of motion is learned according to the knowledge of the plant dynamics. As well-known, in this case it is quite probable that the stability of the closed-loop system cannot be guaranteed, because local stability does not guarantee global stability. This fact implies that the flat hierarchy of learning functions and learned knowledge is necessary in stabilization problems to avoid failure of control.

Unfortunately, although it is already recognized that the order of presenting learned knowledge can influence the behavior of an intelligent system [Lan95], the problem of how to organize the hierarchy of the learning functions and the learned knowledge have received much less attention than they deserve in the existing cognitive architectures. In most cases, such principles are rigidly integrated in the existing cognitive architectures to satisfy some specific contexts, which can be seen implicitly from the unreplaceable topology of different architectures components. Therefore, if a cognitive architecture with sequential organization of knowledge extracting function is used in the stabilization problem, due to the fact it has no capability to change the organization of knowledge extracting functions, it may probably not reach the goal of successful stabilization.

On the other hand, the argument given above does not lead to the conclusion that the issue of stabilization cannot be addressed with the existing cognitive architectures. However, considering that a cognitive architecture should demonstrate generality and flexibility irrelative to implementation domains, it would be desired to relax the rigidity in existing architectures by endowing them with the self-organizing ability of explicitly arranging the interrelationship among cognitive functions and the hierarchy of knowledge.

3.1.2 The proposed architecture

In this contribution, a new cognitive architecture is proposed which is suitable for the stabilization problem. Similar to the human cognition model [Cac98], the proposed architecture is composed of six modules: perception, interpretation, planning,

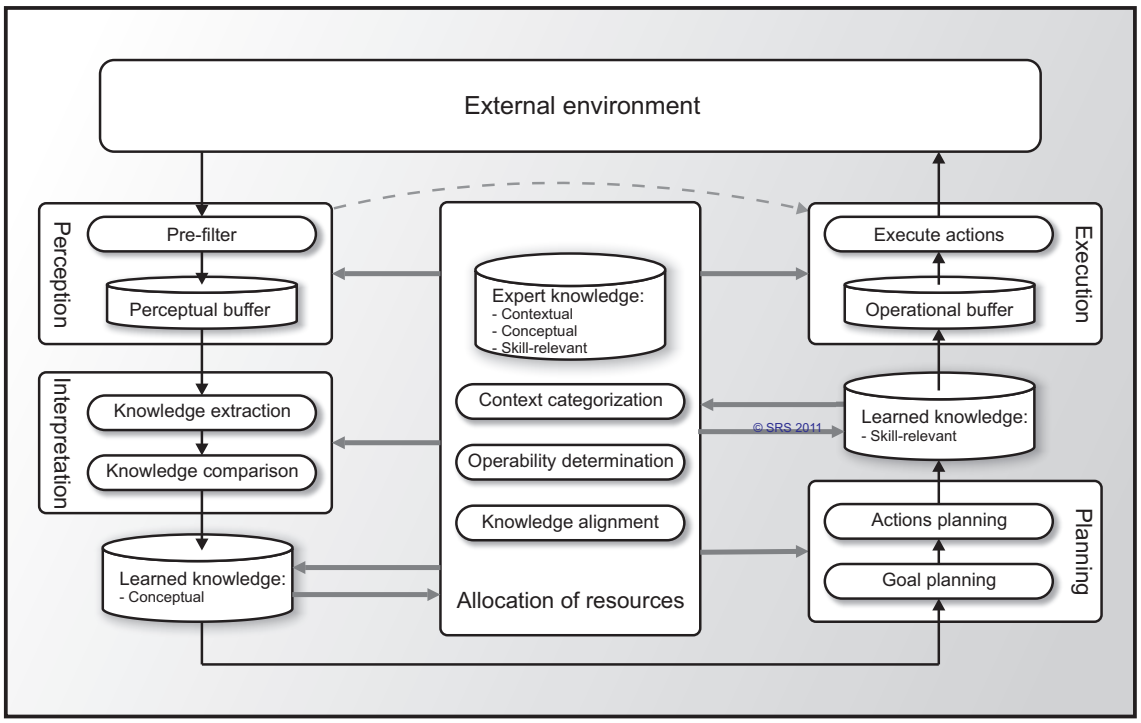


Figure 3.1: Proposed cognitive architecture

execution, allocation of resources, and the knowledge base that is divided into expert knowledge base and learned knowledge base. The scheme of the proposed architecture is shown in figure 3.1.

In the proposed architecture, the module of *Allocation of Resources* (AoR), whose name is borrowed from the human cognition model and originally defined as "defining the level of operability of the cognitive functions and their relative sequence"and "organizing the amount and type of knowledge available for the cognitive process" [Cac98], is used to realize the aforementioned functionality of aligning cognitive functions and corresponding knowledge to guarantee the global goal. All through the complete interaction cycle, the actions of every module are monitored and guided by the AoR module. Because expert knowledge represents the meta understanding of the context settings and problem solving skills and the other functions in AoR cannot perform without these knowledge, the expert knowledge base is also placed in the AoR module. Based on both the learned knowledge and the expert knowledge, the functions in AoR module modify the context categories, determine the operabilities of actions, and guide the planning module to plan the suited goal and the corresponding actions.

The interaction cycle between the cognitive architecture and the external world begins from the perception module. Different from many cognitive architectures which take perception process as barely passive signal measuring, such as EPIC [KWM97],

the perception here is an active process. The raw data is selectively pre-filtered according to the guidance from the contextual expert knowledge but interpreted by the AoR module and structured into the perceptual data buffer.

Secondly, based on the data stored in the perceptual buffer, the representation of the concurrent outside world and the understanding towards this representation are exploited and compared for hierarchial storage, which is the main task of interpretation module. Here the guidance comes from two aspects: the AoR module which defines the hierarchy of learning functions based on the contextual knowledge in the expert knowledge base, and the expert knowledge base which provides the interpretation module with detailed methods of knowledge extraction.

After the work of interpretation being accomplished, the extracted interpretations of the concurrent external world are stored according to the organization structure generated by the AoR module into the learned knowledge base as conceptual knowledge. The AoR module also defines the presenting sequence of this learned conceptual knowledge to the planning module.

With the learned conceptual knowledge in the learned knowledge base, the planning module arranges the goal for this interaction cycle and generate action plans, if AoR cannot find suitable skills from the learned knowledge base of skill-relevant knowledge. The planning methods are provided by the expert knowledge with expert skill-relevant knowledge. Although the planned goal is local if the learned knowledge is local, the AoR module defines the mechanisms guiding the sequence of these goals at every interaction cycle and the corresponding actions, to guarantee the global goal can be reached. The generated action plan is stored in the learned knowledge base of skill-relevant knowledge. If AoR can find a suitable skill in the learned knowledge base, this step is overleaped and the skill stored in the learned knowledge base is utilized directly as action plan.

Finally, the generated action plan is sent to the operational buffer and executed by the execution module. The execution methods is also influenced by the AoR according to its context categorization. This complete one cycle of interaction between the cognitive agent and its external world.

The proposed architecture also satisfies the three-level hierarchy of cognitive control shown in figure 1.1. The dashed line in figure 3.1 connects the perception and execution and represent the control without learning ability. If the expert knowledge base and the AoR module are removed, the left structure represents control with learning ability and pre-defined working patterns, whereas the AoR module and the expert knowledge base stand for the deliberative level, control with learning ability and self-organized working patterns.

Because the online identification of unknown system dynamics can only be realized in a incremental learning fashion, the proposed cognitive architecture can be utilized to construct the cognition-oriented control solving stabilization of unknown systems, which is introduced in the next section of this chapter.

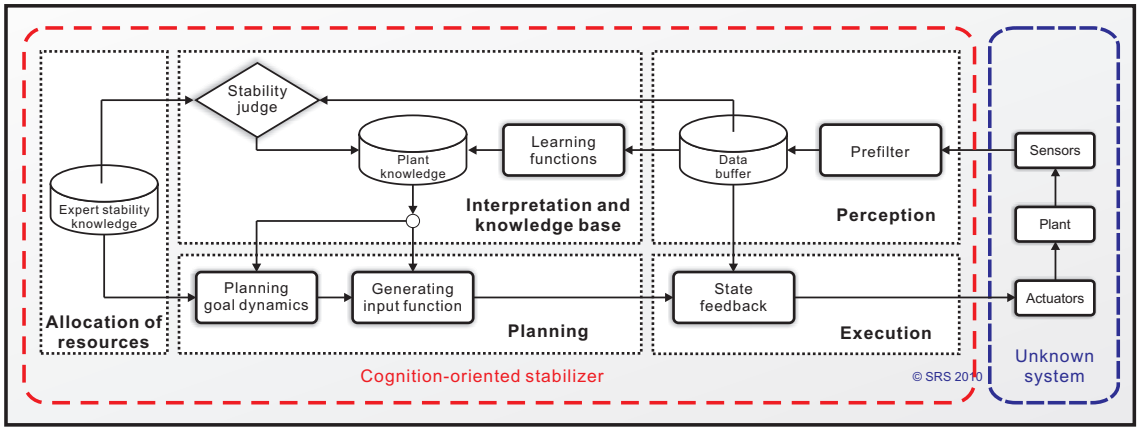


Figure 3.2: Framework of cognition-oriented stabilization

3.2 Realization of cognition-oriented stabilization

As stated in section 1.4, the stabilization problem considered in this contribution is to find such a suitable input function $\boldsymbol{u}(k) = \boldsymbol{u}(\boldsymbol{x}(k), \boldsymbol{x}(k-1), \ldots, \boldsymbol{x}(k-l))$ that the nonlinear discrete-time system

$$\boldsymbol{x}(k+1) = \boldsymbol{f}(\boldsymbol{x}(k), \boldsymbol{u}(k)), \tag{3.1}$$

is stable, with the same assumptions in the proposition of the data-driven stability criterion: the nonlinear function $\boldsymbol{f}(\cdot)$ is unknown; system states are fully measurable; and the measurements are noisy free.

According to the proposed cognitive architecture, the framework of cognition-oriented stabilization is established, as shown in the figure 3.2.

In this framework, the unknown system to be controlled can be taken together as an unknown environment. The stimuli to the cognitive agent are the signals from sensors measuring input and output of the plant, and the response is the control input signal transferred to actuators of the plant. The task of cognition is to realize the goal of stabilization by learning the plant dynamics $\boldsymbol{f}(\cdot)$ and automatically defining and realizing a suitable control input function $\boldsymbol{u}(k)$.

The interconnections among different parts in the proposed architecture are simplified because of the clearance of stabilization problem setting. Because the goal of control is usually clear defined in advance, the context of the control task is already clear and needs not to be classified in the cognition processing. As a result, it should be noted that in this framework only expert knowledge base is kept in the AoR module and the functions in the AoR in the architecture are not shown. Instead, the rules generated by these functions are represented by the topology of the framework components and some of the expert stability knowledge. The expert knowledge base

contains the conceptual and the skill-relevant knowledge about stability, which can guide the interpretation module to obtain the characteristics of the plant useful for the aim of control, and help the planning module to generate suitable goal dynamics.

Every block in the framework serves as a module with its specific functionality and can be realized in different methods. For example, the functionality of determining the concurrent plant dynamics can be realized by both neural-network-based system identification technology and the incremental model tree induction. However, the realization of each module must be chosen in a consistent and proper way such that no communication problems among different modules can be encountered. The following parts of this section introduce one possible realization of each blocks in the framework and how they work together to solve the concerned stabilization problem.

3.2.1 Perception and execution

If there exist only the perception module and the execution module that are linked directly, a controller without learning capability can be formulated. Therefore, the realization of these two modules is put together to explain the lowest level within the three-level hierarchy of the proposed cognition-oriented stabilization method.

In the discussion about the proposed architecture it is shown that the perception functions have the duty on separating and structuring information from raw sensor data in order to satisfy the demand of further cognitive processing. In this realization of the proposed framework, online system identification methods are utilized to discover plant dynamics. Thus the input-output data of the plant are required, which can be gathered directly from sensors.

Further more, because the expert knowledge about stability requires a full measurement of system states, the orders of the measured data should be determined by the pre-filter properly such that the identified input-output model can be represented into state-space representation. This requirement represents de facto the guidance of the AoR module in the proposed architecture. After that, the selected data are structured and stored in the data buffer waiting for further cognitive processing.

The proposed stability criterion, as part of the expert knowledge, requires that the concerned system should be represented in the form of $\boldsymbol{x}\,(k+1) = \boldsymbol{f}(\boldsymbol{x}\,(k))$. Therefore, the control input should have a form of state feedback, as $\boldsymbol{u}(k) = \boldsymbol{u}(\boldsymbol{x}(k))$, so that the closed-loop system $\boldsymbol{x}(k+1) = \boldsymbol{f}(\boldsymbol{x}(k), \boldsymbol{u}(k))$ is compatible with requirement to use the expert stability knowledge. As it is assumed that all the system states are measurable, the input function can be further detailed into the state feedback form, as

$$\boldsymbol{u}(k) = -\boldsymbol{K}(k)\boldsymbol{x}(k), \tag{3.2}$$

where $\boldsymbol{K}(k)$ is the feedback matrix whose elements can be changed at different time instant to produce different values of control input. It can be seen that if there is no other blocks in the framework and the matrix $\boldsymbol{K}(k)$ is known, the control degenerates to the standard state-feedback form without any intelligent abilities.

3.2.2 Interpretation and learned knowledge base

The interpretation part should produce two kinds of dynamical knowledge: a dynamic model of the plant and the stability judgement of the current closed-loop system motion.

The learning function used for system identification should be able to used in an online context. The representation of the identified dynamics should be in the same form as the representation (3.1), denoted as

$$\hat{\boldsymbol{x}}(k+1) = \hat{\boldsymbol{f}}(\boldsymbol{x}(k), \boldsymbol{u}(k)), \tag{3.3}$$

where $\hat{\boldsymbol{f}}(\cdot)$ represents the identified plant dynamics, $\hat{\boldsymbol{x}}(k+1)$ represents the one-step prediction of the future states of the plant at the time instant $t = k$. At every time constant, the new plant input and output data is used to train the model, so that with the number of interactions between the controller and the plant increasing, the precision of the identified model should be improved incrementally, and so does its prediction of the (local) future plant response, which means $\hat{\boldsymbol{f}}(\boldsymbol{x}(k), \boldsymbol{u}(k)) \rightarrow \boldsymbol{f}(\boldsymbol{x}(k), \boldsymbol{u}(k))$ if $k \rightarrow \infty$ and the one-step prediction error of the identified dynamics defined as

$$\boldsymbol{e}_m(k+1) = \hat{\boldsymbol{x}}(k+1) - \boldsymbol{x}(k+1), \tag{3.4}$$

follows

$$\lim_{k\rightarrow\infty} \parallel \hat{\boldsymbol{x}}(k+1) - \boldsymbol{x}(k+1) \parallel \rightarrow \boldsymbol{0}, \tag{3.5}$$

where $\parallel \cdot \parallel$ represents the norm calculation. It does not matter which system identification technique is utilized to model the plant dynamics, such as the fuzzy logic method [Lil10], or the online identification using Recurrent Neural Networks (RNN) [WZ89] which is adopted in the first numerical example of this thesis in fourth chapter (the training mechanism of RNN is shown in figure 3.3 with $\boldsymbol{u}(k)$ being the input, $\boldsymbol{x}(k)$ and $\boldsymbol{x}(k+1)$ the system states, and $\hat{\boldsymbol{x}}(k+1)$ the estimated states by RNN). As long as the above requirements towards the learning of the plant dynamics can be satisfied, the different system identification methods would not influence the results of stabilization but only the performance of the control such as the converging speed and so forth.

As introduced in the first section in this chapter, these two kinds of knowledge should be arranged in a flat manner. As a result, the corresponding two cognitive

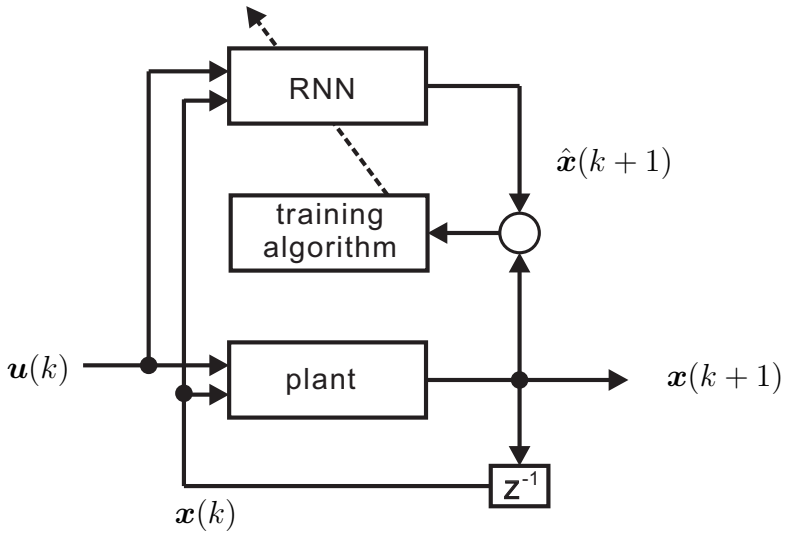


Figure 3.3: Working mechanism of online system identification using RNN [WZ89]

functions, the learning function used for system identification and the stability judge based on expert knowledge, are arranged in parallel, i.e., the data store in the data buffer in the perception module is sent to these two functions in parallel, so that the incremental learning existed in the learning functions of system identification does not influence the correctness of the stability judgement. The parallel arrangement reflects the guidance of AoR in the proposed architecture under the context of stabilization.

From the above discussion it can be seen that because the stability judgment should be made directly from the measured data, the stability judgment should be realized in an online and data-driven manner. As a result, the data-driven stability criterion proposed in the second chapter, i.e. the theorem 2.4.4, is utilized here. In detail, the stability of the current plant motion is represented geometrically into the form of a polyhedral cone $\mathbf{cone}\,\tilde{\mathcal{V}}_{r-1}$ defined in equation (2.32). Please note that this knowledge about stability is also incrementally extracted, as indicated by equation 2.31 in chapter 2. If new data comes, it is only necessary to refine the stability knowledge by calculating the intersection between the old polyhedral cone $\mathbf{cone}\,\tilde{\mathcal{V}}_{r-1}$ and the geometrical constraints defined by the new data about stability defined in equation (2.27). The representation of current plant motion is stored in parallel to the identified plant dynamics in the learned knowledge base for further dynamical processing.

3.2.3 Expert stability knowledge

It is already mentioned that the proposed data-driven stability criterion is the expert knowledge. Indeed, the criterion itself represents only the conceptual expert knowl-

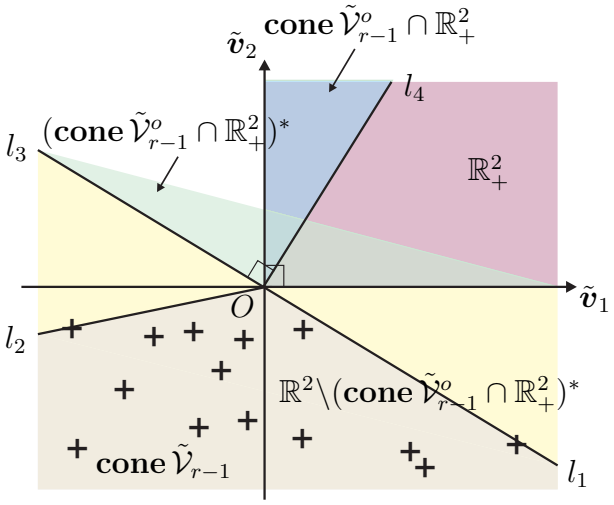


Figure 3.4: The skill-relavent expert knowledge about stability

edge. It needs to be reformulated in a different manner to serve as the skill-relevant expert knowledge that can be used to guide the planning module to generate suitable actions.

The skill-relevant expert knowledge about stability answer the question whether the predicted states lead to stable dynamics. The two-dimensional representation of this knowledge is shown in figure 3.4.

At the time instant $t = r$ the r times measurements of system states are transformed into $r-1$ vectors according to the equation (2.24) in the second chapter, denoted as $\tilde{\boldsymbol{v}}$ and shown as crosses in figure 3.4. The polyhedral cone determined by these vectors is denoted as $\mathbf{cone}\,\tilde{\mathcal{V}}_{r-1}$. In figure 3.4, this polyhedral cone $\mathbf{cone}\,\tilde{\mathcal{V}}_{r-1}$ is shown as the yellowgrey area between the two rays l_1 and l_2.

According to the theorem 2.4.4 about the data-driven quadratic stability, the concerned system is quadratic stable if at any time the intersection between the polar cone of $\mathbf{cone}\,\tilde{\mathcal{V}}_{r-1}$, denoted as $\mathbf{cone}\,\tilde{\mathcal{V}}^o_{r-1}$, and the positive real space $\mathbb{R}^2_+$, is not empty. In fact, the intersection $\mathbf{cone}\,\tilde{\mathcal{V}}^o_{r-1}\cap\mathbb{R}^n_+$ represents the solution to the inequality set (2.27), which can be seen from the deducing process from the theorem 2.4.4.

In figure 3.4, the area between the positive directions of the two coordinates represent the positive real space $\mathbb{R}^2_+$ and painted with violet color, some part of which is covered by the blue. The intersection between $\mathbf{cone}\,\tilde{\mathcal{V}}^o_{r-1}$ and $\mathbb{R}^2_+$, denoted as $\mathbf{cone}\,\tilde{\mathcal{V}}^o_{r-1}\cap\mathbb{R}^2_+$, is represented as the lightblue area between the positive direction of coordinate $\tilde{\boldsymbol{v}}_2$ and the ray l_4. According the proposed stability criterion, the system is quadratic stable if the interior of the intersection $\mathbf{cone}\,\tilde{\mathcal{V}}^o_{r-1}\cap\mathbb{R}^2_+$ is not empty, which means the blue area in figure 3.4 is not empty.

At the next time instant $t = r + 1$, a new vector $\tilde{\boldsymbol{v}}(r)$ can be obtained if the state vector $\boldsymbol{x}(r + 1)$ is known. Under this circumstances, the concerned system can be judged as quadratic stable, if the condition defined in the inequality set (2.27) can be satisfied, which means there exist at least one common solution of the old inequality set (2.27) with $k = r$ and a new inequality

$$\langle \tilde{\boldsymbol{v}}(r), \boldsymbol{d} \rangle \leq 0, \quad \text{with} \quad \boldsymbol{d} \in \mathbb{R}^n_+. \tag{3.6}$$

This fact indicates that if such a solution exist, it is located within the intersected set $\mathbf{cone}\, \tilde{\mathcal{V}}^o_{r-1} \cap \mathbb{R}^n_+$ constructed at $t = r$. Therefore, the concerned system can be judged as stable at $t = r + 1$ if and only if the transformed vector $\tilde{\boldsymbol{v}}(r)$ is not located in any region the vectors in which have positive inner products with an arbitrary vector located in the old intersection $\mathbf{cone}\, \tilde{\mathcal{V}}^o_{r-1} \cap \mathbb{R}^n_+$ constructed at $t = r$. Otherwise the inequality (3.6) cannot have a common solution with the old inequality set (2.27) with $k = r$.

Recalling the definition of dual cone defined by equation (2.11) in the second chapter, it would be found that the unstable region for the new vector $\tilde{\boldsymbol{v}}(r)$ is exactly the dual cone of the intersection $\mathbf{cone}\, \tilde{\mathcal{V}}^o_{r-1} \cap \mathbb{R}^n_+$ constructed at $t = r$, which is denoted as $(\mathbf{cone}\, \tilde{\mathcal{V}}^o_{r-1} \cap \mathbb{R}^n_+)^*$ and shown in the two-dimensional case in figure 3.4 as the green area between the ray l_3 and the positive direction of the coordinate $\tilde{\boldsymbol{v}}_1$.

As a result, from the above discussion it can be concluded that the system states $\boldsymbol{x}$ at next time step in future can lead to a stable dynamics if and only if its corresponding transformed vector $\tilde{\boldsymbol{v}}$ is located in the complementary set of the dual cone $(\mathbf{cone}\, \tilde{\mathcal{V}}^o_{r-1} \cap \mathbb{R}^n_+)^*$, which is denoted as $\mathbb{R}^n \backslash (\mathbf{cone}\, \tilde{\mathcal{V}}^o_{r-1} \cap \mathbb{R}^2_+)^*$ and shown in the two-dimensional case in figure 3.4 as the yellow area with the cone $\mathbf{cone}\, \tilde{\mathcal{V}}_{r-1}$ included. Please note that this region feasible for stable system states is not a convex set. If the system states at next time step is always located in the region $\mathbb{R}^n \backslash (\mathbf{cone}\, \tilde{\mathcal{V}}^o_{r-1} \cap \mathbb{R}^2_+)^*$, the quadratic stability of the system can be guaranteed. This fact provides the possibility for the planning module to know how to evaluate whether an input candidate can be used to stabilize the system by examining the system states generated by the input candidate is located in the feasible region $\mathbb{R}^n \backslash (\mathbf{cone}\, \tilde{\mathcal{V}}^o_{r-1} \cap \mathbb{R}^2_+)^*$.

3.2.4 Planning situated actions

The form of execution has already been defined as state feedback with a varying feedback matrix $\boldsymbol{K}(k)$, as shown in equation (3.2). Therefore, the task for planning is how to choose the suitable values of the elements of $\boldsymbol{K}(k)$ such that a globally stable dynamical behavior can be achieved.

Defining goals of control

Although to obtain stable closed-loop dynamics is the main goal of cognition-oriented stabilization, this goal cannot be used to generate a detailed input function, because there exists not only one suited solution to the stabilization problem. Therefore, an additional goal has to be integrated into the planning mechanism for selecting suitable control input and requirement of stability is here taken as a constraints for the selection.

This additional goal can be chosen arbitrarily according to the preference of the context requirements. It could be to minimize the distance between the current states to the desired states in the state space, or a performance measure evaluating the functionality of the current control. However, the most important sense of this goal lies in that it can generate a corresponding criterion which can be used to choose one suitable control input within the stability constraints.

One possible realization is to take the evaluation of the control performance with respect to input energy and control errors as such a goal for planning, which and can be defined as

$$J = \int_{t_0}^{t_1} \boldsymbol{x}^T \boldsymbol{E} \boldsymbol{x} + \boldsymbol{u}^T \boldsymbol{F} \boldsymbol{u} \, \mathrm{d}t, \tag{3.7}$$

where t_0 and t_1 are the starting time and the final time of the evaluation, $\boldsymbol{E}$ and $\boldsymbol{F}$ are positive definite matrices, and $\boldsymbol{x}$ and $\boldsymbol{u}$ are the system states and control inputs, respectively. This ratio between the norms of $\boldsymbol{E}$ and $\boldsymbol{F}$ represents the compromise between the integral squared error of control and the input energy.

At every time instant, an optimization problem similar to the linear optimal control can be established with respect to the performance function above. The elements of the feedback matrix $\boldsymbol{K}$ can be taken as the optimization variables and the stability criterion can be utilized as the optimization constraints. After each optimization a suitable matrix $\boldsymbol{K}$ can be found and applied as the control input function to the plant.

As previously stated, within this thesis the goal of control performance is not the primary goal, and it must compromise to the principal goal of stabilizing the target system. Therefore, it is not necessary for the planning module to use this strategy to obtain a globally optimal control with respect to the performance measure (3.7), but functions as the selecting measure to obtain one certain control input which can be used to achieve the primary goal.

It should be mentioned that the global stability is guaranteed by the data-driven stability criterion, which takes the stability of the current motion into consideration, instead of the stability judgment of the identified model. This fact makes the proposed method different from the Model Predictive Control (MPC) and avoids the stability issue of MPC which is difficult to deal with.

Planning goal dynamics and generating input function

A cognitive system should be able to adjust its functionalities to fulfill the requirement of the given primary goal. In the proposed stabilization method, this ability is realized by formulating a meta representation of the goal dynamics according to the obtained knowledge, and by adjusting it with respect to new situations. The control input is properly selected to follow the formulated goal dynamics.

Here, the identified model is not used in the proposed method to predict the plant states at next time instant, but to plan the goal dynamics that the plant should follow. In other words, the optimization problem mentioned above to select input functions is not used directly to plan the states of the plant, but the states of the goal dynamics.

Consider the identified model given in (3.3). At an arbitrary time instant $t = r$, define the set of the estimated states $\hat{\boldsymbol{x}}(k)$, $k = 1, \ldots, r, r+1$, as $\hat{\mathcal{X}}_{r+1}$, where the last state $\hat{\boldsymbol{x}}(r+1)$ is obtained by applying the input $\boldsymbol{u}(k)$ with one certain feedback matrix $\boldsymbol{K}$ to the rebuilt plant dynamics $\hat{\boldsymbol{x}}(k+1) = \hat{\boldsymbol{f}}(\boldsymbol{x}(k), \boldsymbol{u}(k))$. The corresponding transformed vectors according to the equation (2.24) as $\boldsymbol{w}(k)$, and the convex cone determined by the transformed vectors as $\mathbf{cone}\,\mathcal{W}_r$. Then the optimization problem for seeking control input at $t = r$ can be stated as follows

$$\begin{aligned} \min_{\boldsymbol{K}} \quad & J \\ s.t. \quad & \boldsymbol{u}(k) = -\boldsymbol{K}\boldsymbol{x}(k), \\ & \mathbf{cone}\,\mathcal{W}_r^o \cap \mathbb{R}_+^n \neq \varnothing. \end{aligned} \tag{3.8}$$

The optimization problem above is indeed a very complicated problem with the complex nonlinear constraints, the existence of whose solution cannot be analytically determined. If the solution space (here the varying domain of the elements of $\boldsymbol{K}$) is not too large, this problem can also be solved by searching methods, i.e., to search the solution space and find the set of elements of $\boldsymbol{K}$ which can leads to the smallest value of J. Because the optimality here is not the main goal, the suboptimal solution is also acceptable.

An example of the possible algorithms to find the suboptimal solution to the optimization problem (3.8) is shown as in algorithm 1. In this example algorithm, a finite set of the feedback gain matrix $\boldsymbol{K}$ is given and denoted as Ξ. The example algorithm calculates the value J of the cost function defined in the optimization problem (3.8) for every $\boldsymbol{K}$ in Ξ and finds the one which possesses the smallest J and can by the meantime satisfy the stability condition. By this way, a suitable feedback gain $\boldsymbol{K}$ which can stabilize the plant at the time instant $t = r$ is found. Although this $\boldsymbol{K}$ is only optimal for the present time instant, it can be applied by the execution module to stabilize the system to be controlled.

input : current system state $\boldsymbol{x}(k)$;
searching region Ξ for feedback gain matrix $\boldsymbol{K}$;
identified plant dynamics $\hat{\boldsymbol{x}}(k+1) = \hat{\boldsymbol{f}}(\boldsymbol{x}(k), \boldsymbol{u}(k))$;
data set $\hat{\mathcal{X}}_k$ with $\hat{\mathcal{X}}_k = \{\hat{\boldsymbol{x}}(0), \hat{\boldsymbol{x}}(2), ..., \hat{\boldsymbol{x}}(k)\}$;

output : feedback gain matrix $\boldsymbol{K}$;

initialize: set value $i \leftarrow 0$, $j \leftarrow 0$;
set value $J \leftarrow$ 1e20;
set value n ; `// n: number of elements of Ξ`

while $i \leq n$ **do**
  set value $\boldsymbol{K} \leftarrow \boldsymbol{K}_i$ with $\boldsymbol{K}_i \in \Xi$;
  calculate $\boldsymbol{u}_i(k) = -\boldsymbol{K}\boldsymbol{x}(k)$;
  calculate $\hat{\boldsymbol{x}}_i(k+1) = \hat{\boldsymbol{f}}(\boldsymbol{x}(k), \boldsymbol{u}_i(k))$;
  for $j \leftarrow 1$ **to** k **do**
    calculate $\hat{\mathcal{X}}_{k+1}$ with $\hat{\boldsymbol{x}}(k+1)$;
    calculate $\boldsymbol{w}(j)$; `// according to the equation (2.24)`
  end
  calculate **cone** $\mathcal{W}_r^o$; `// establish the polar cone`
  if cone $\mathcal{W}_r^o \cap \mathbb{R}_+^n \neq \varnothing$ **then**
    calculate J_i;
    if $J_i < J$ **then**
      calculate $J = J_i$
    end
    calculate $i = i + 1$;
  else
    calculate $i = i + 1$;
  end
end

Algorithm 1: Solving optimization problem (3.8) by complete search

Recalling equation (3.4) which indicates that the error $\| \boldsymbol{e}_m \|$ between the estimated states and the real states is incrementally decreasing to zero, the solution to the optimization problem (3.8), denoted as $\boldsymbol{K}(k)^*$, can be directly applied to generate the control input, as $u(k) = -\boldsymbol{K}(k)^*\boldsymbol{x}(k)$. The reason is that the estimated states $\hat{\boldsymbol{x}}(k)$ satisfies the data-driven stability criterion, shown as the second constraint in the optimization problem (3.8), the sequence of the estimated $\hat{\boldsymbol{x}}(1)$, $\hat{\boldsymbol{x}}(2)$, ..., $\hat{\boldsymbol{x}}(k)$, ... represent a stable dynamics. Because the error $\| \boldsymbol{e}_m \|$ is approaching zero with time going, the plant dynamics would become the same as the stable dynamics of the estimated states. Due to this reason, the estimated states sequence can be treated as the goal dynamics that the plant should follow. By this way the suitable

control input can be generated and applied to the plant to realize cognition-oriented stabilization.

3.3 Summary of this chapter

In this chapter, a cognitive architecture which focus on the alignment of the learning function and the learned knowledge, as well as the guidance of the contextual knowledge on different cognitive functions is firstly introduced. Because its self-organizing mechanism of the learning functions, the local stability issue which is often encountered in model predictive control can be avoided. After that, a framework used for cognition-oriented stabilization is proposed based on the data-driven stability criterion in last chapter and the proposed cognitive architecture. and one of its detailed realization strategy is introduced. After reformulating the proposed stability criterion as the skill-relevant expert knowledge, a local optimization process is proposed to use the identified model to plan a stable goal dynamics and to generate the corresponding control input that driving the plant to approach the stable goal dynamics, whereby the global stabilization is realized.

4 Numerical Examples

In this chapter, the proposed cognition-oriented control used for stabilizing unknown discrete-time system is utilized in two simulation examples: an introducing example with a pendulum system to demonstrate the detailed realization process, and an application example of stabilizing a nonlinear aeroelastic system.

4.1 Introducing examples with a pendulum system

4.1.1 Task description

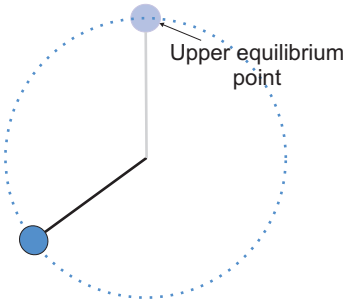


Figure 4.1: The pendulum system

The pendulum system is a classic nonlinear system used in many examples for stability analysis and control. The configuration of the pendulum system is shown in figure 4.1. In this contribution, the equation of motion the considered pendulum system is shown as follows

$$\begin{cases} \dot{x}_1(t) &= x_2(t), \\ \dot{x}_2(t) &= -10\sin\left(x_1(t)-\pi\right) - x_2(t) + u(t). \end{cases} \tag{4.1}$$

where x_1 and x_2 are the system states which represent the position and the angular velocity of the pendulum, respectively, and $u(t)$ represents the control input. The goal of the control is to stabilize the pendulum system at its upper equilibrium point with the assumption that the system dynamics is unknown and the system states are fully measurable without noise.

4.1.2 Controller determination

According to the introduction to the realization of the proposed cognition-oriented controller, it is necessary to choose one certain system identification method to serve

as the learning function and a measure of system functionalities to help finding the suitable control input.

In this pendulum example, the measure of the system functionality is chosen as the measure of the integral squared error of control, defined as

$$J = \int_{t_k}^{t_{k+1}} \boldsymbol{x}^T \boldsymbol{E} \boldsymbol{x} \, \mathrm{d}t, \tag{4.2}$$

where matrix $\boldsymbol{E}$ is a diagonal positive definite matrix. In this contribution, the diagonal element vector of $\boldsymbol{E}$ are chosen as $[q\ 1]^T$, where q is a positive real number.

The Recurrent Neural Network (RNN) [WZ89], which can be used for online black-box system modeling, is utilized to serve as the system identifier. The choice of these two representation are made to satisfy the requirements of learning functions and the functionality measure, but without any preference. They could be replaced by other techniques which satisfies these requirements introduced in chapter 3.

With the system functionality measure and the learning function to extract the plant dynamics being settled, the feedback matrix $\boldsymbol{K}(k)$ can be determined by solving the optimization problem (3.8) by replacing J with defined functionality measure and $\hat{\boldsymbol{x}}(k+1)$ with the estimated value of $\boldsymbol{x}(k+1)$ by RNN. By solving this optimization problem at every time instant and apply the control input to the pendulum system, the pendulum should be stabilized at the upper equilibrium point.

The structure of RNN

The RNN system identification is a well-developed online system identification method originating from the beginning of 1980s. The network structure of RNN is shown in figure 4.2. The network consists of two distinct layers: a concatenated input–feedback layer and a processing layer of computation nodes (sometimes called neurons) [Hay99]. The inputs to the input-feedback layer contains the fed back state vector $s(k)$ with one step delays and the external input vector $\hat{\boldsymbol{u}}(k)$. The outputs of the input-feedback layer are the inputs to the processing layer. Only part of the outputs of the neurons are taken as the outputs of the whole network, shown as $\hat{\boldsymbol{y}}(k+1)$ in the figure 4.2. The dynamics of RNN can be represented as

$$\begin{cases} \boldsymbol{s}(k+1) &= \boldsymbol{h}(\boldsymbol{W}_s \boldsymbol{s}(k) + \boldsymbol{W}_{b\hat{d}u} \hat{\boldsymbol{u}}(k)) \\ \hat{\boldsymbol{y}}(k) &= \boldsymbol{C}\boldsymbol{s}(k), \end{cases} \tag{4.3}$$

where $\boldsymbol{W}_s$ and $\boldsymbol{W}_{b\hat{d}u}$ represent the weights of the vector $\boldsymbol{s}(k)$ and $\hat{\boldsymbol{u}}(k)$ respectively, $\boldsymbol{h}(\cdot)$ the activation function, and $\boldsymbol{C}$ the output matrix.

In this example the widely accepted real-time recurrent learning (RTRL) has been selected as the training algorithm. The RTRL algorithm uses a gradient descent procedure to compute the weight changes and can obtain incrementally the description of plant dynamics [WZ89], which is required by the proposed cognitive stabilization framework.

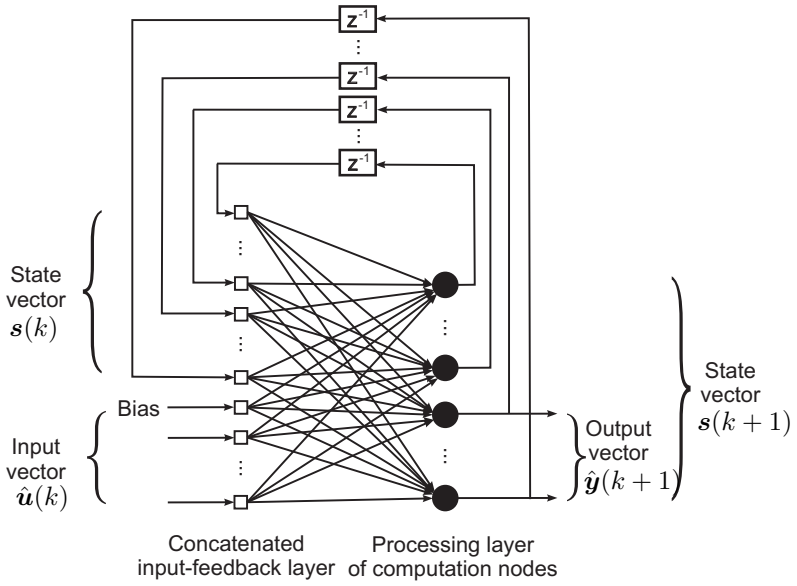


Figure 4.2: Network structure of RNN [Hay99]

4.1.3 Simulation results

The phase portrait of the pendulum system under the proposed control is shown in figure 4.3. In this simulation, four neurons and two external inputs are utilized for the RNN to identify the pendulum dynamics. The initial condition for this simulation is set as $[-2 \quad 5]$, the simulating time as 5 seconds, and the sampling time as 1×10^{-2} second. The optimization problem is solved by searching and the varying interval of the elements of the feedback matrix is defined as $(-5\ , 5)$. Because the RTRL algorithm needs a few time steps to train its weight matrix before its estimations reach a high accuracy, there is a training period at the beginning of the whole control process, which means that there is no control input at the first several time steps. In this simulation, the first 30 time-steps are settled as the training period.

The origin point of the phase portrait denotes the upper equilibrium point and the blue line indicates the open loop response of the pendulum system. It is clear that the system states will go to the lower equilibrium point without control input. The green line indicates the predicted system states by RNN whose initial value is always zero. At the end of the training period (here denoted from the origin point to point A), the weight matrix of the system identifier has been trained to have the capability to predict the real system with a high accuracy. The controller is turned on after the point A. The red line is the actual closed loop response of the system which does approach the origin point step by step using the proposed cognitive stabilizer. The phase portrait near the origin point is pointed out more clearly at the right side.

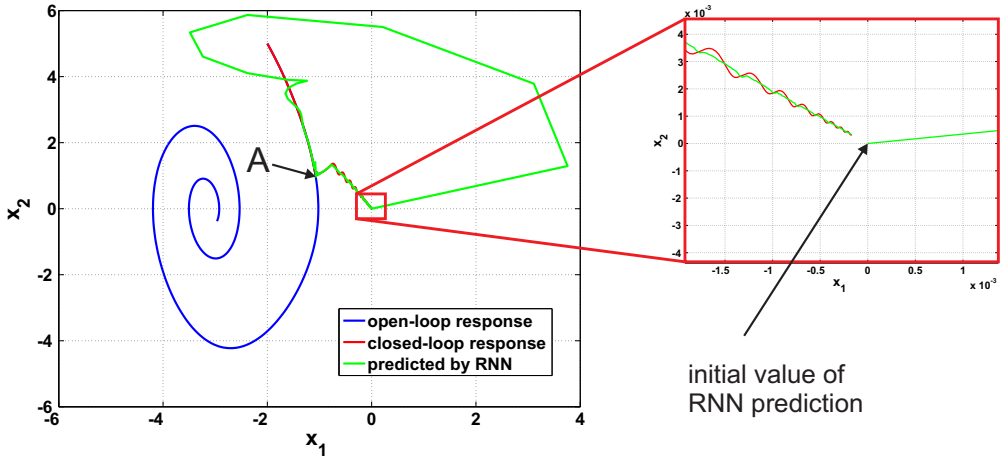


Figure 4.3: Phase portrait of the pendulum controlled by proposed method

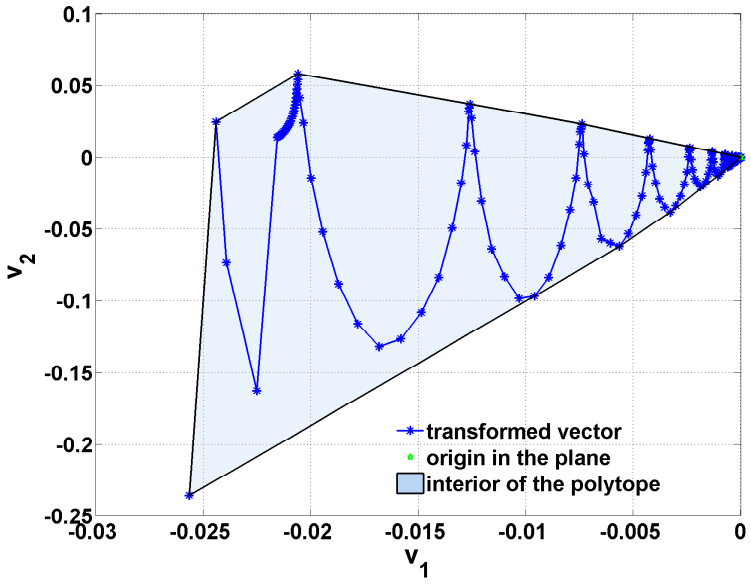


Figure 4.4: Transformed vectors and the corresponding polytope

The result of the data-driven stability judgement at the 5-th second is shown in figure 4.4. Obviously, the smallest convex cone containing the polytope (shown as the lightblue area in the figure 4.4) is located in a negative half space, which means that the motion of the system from the beginning of control until $t = 5$s can be judged as quadratic stable.

The time history of each system states, i.e., the position and the velocity of the pendulum, the control input, and both state feedback gains are shown in the figure 4.5.

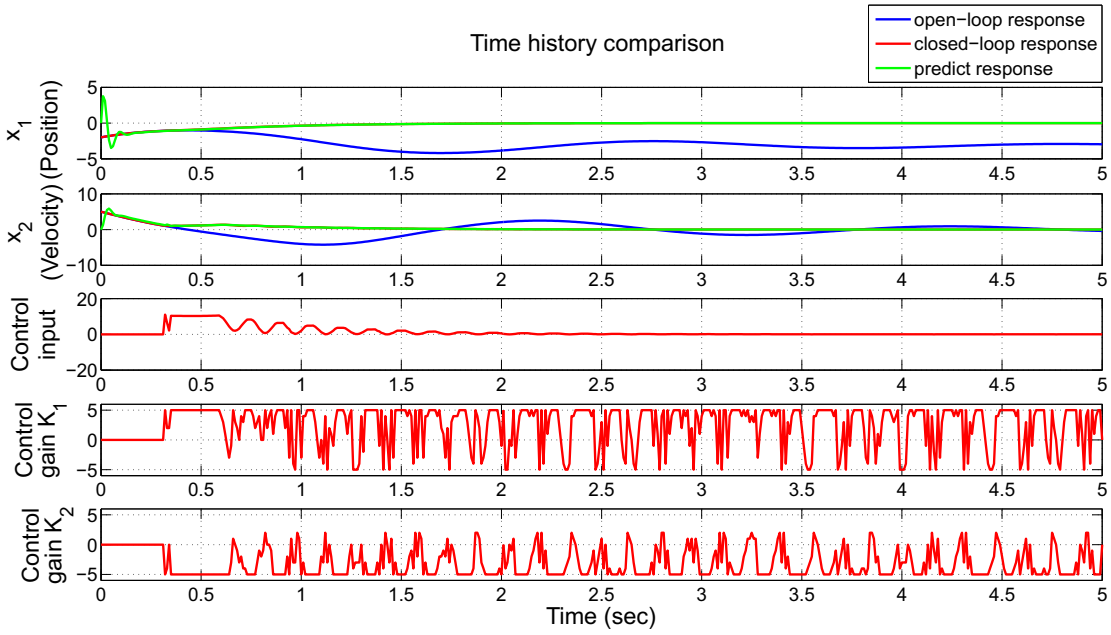


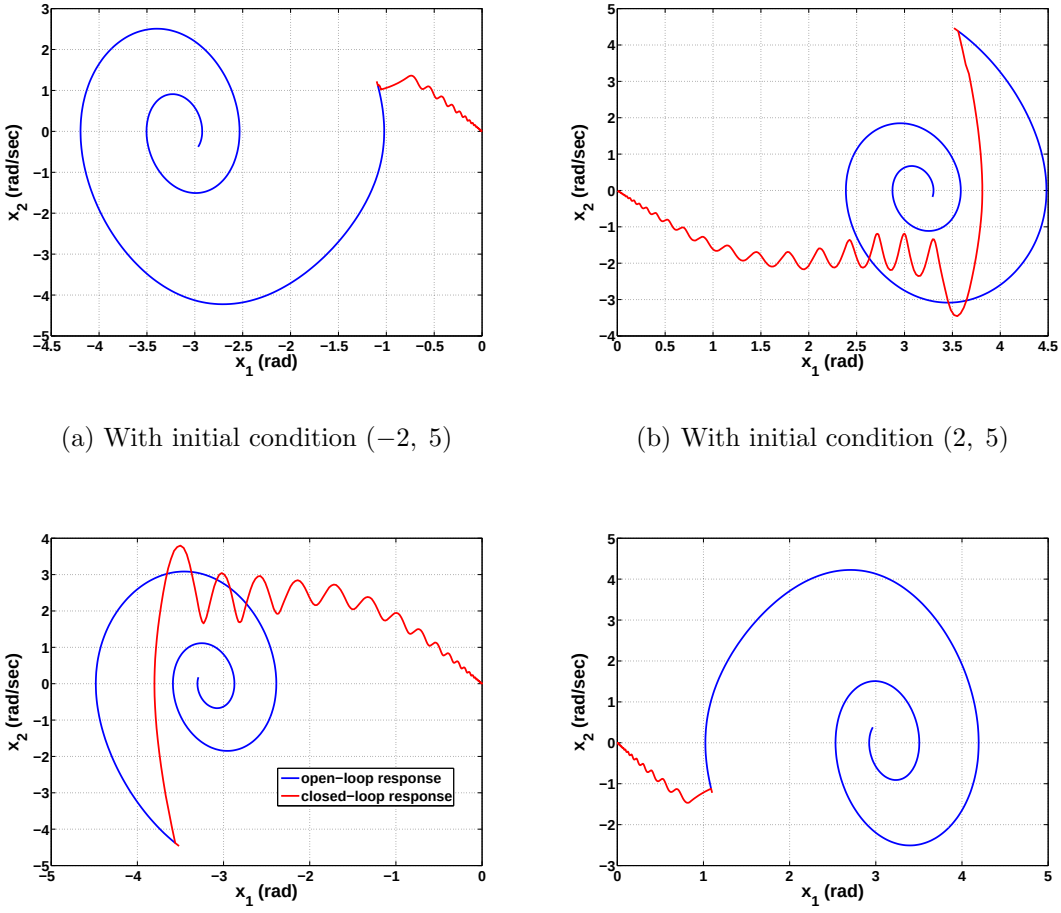
Figure 4.5: Time history of the closed-loop system

It can be seen from this figure that after 1.5 seconds, the position and the velocity of the system almost converge to zero. In addition, the system control input decreases to zero determined by the state feedback gains which can always be found in the whole process. Therefore, the proposed cognitive stabilizer can be used to achieve the control goal successfully.

In order to check the adaptivity of the proposed cognitive stabilizer to different initial conditions, two couples of symmetrical initial points such as $(2,\ 5)$ and $(2,\ -5)$, $(-2,\ 5)$ and $(-2,\ -5)$ are tested in the following. The simulation results are shown in the figure 4.6. The first 30 steps of training phase of the RNN are not shown; only the phase trajectories after the control is turned on are shown. It can be seen from figure 4.6 that the pendulum system under control has almost symmetrical dynamic responses with respect to symmetrical initial conditions, which implies that proposed control strategy can produce reliable and adaptive performances.

It should be pointed out that the simulation results with respect to symmetric initial conditions are not completely symmetric. The reason is because of the different trained weight matrices for when control is turned on. Although the initial weight matrices has been selected for the symmetrical initial points, the weight matrices are still not identical after the training period because the initial values of the two weight matrices are not symmetrical with respect to the initial conditions.

Similar simulation results can be obtained under the initial conditions changed with the same scaling. After changing the scale of all the four initial conditions to



(a) With initial condition $(-2,\ 5)$

(b) With initial condition $(2,\ 5)$

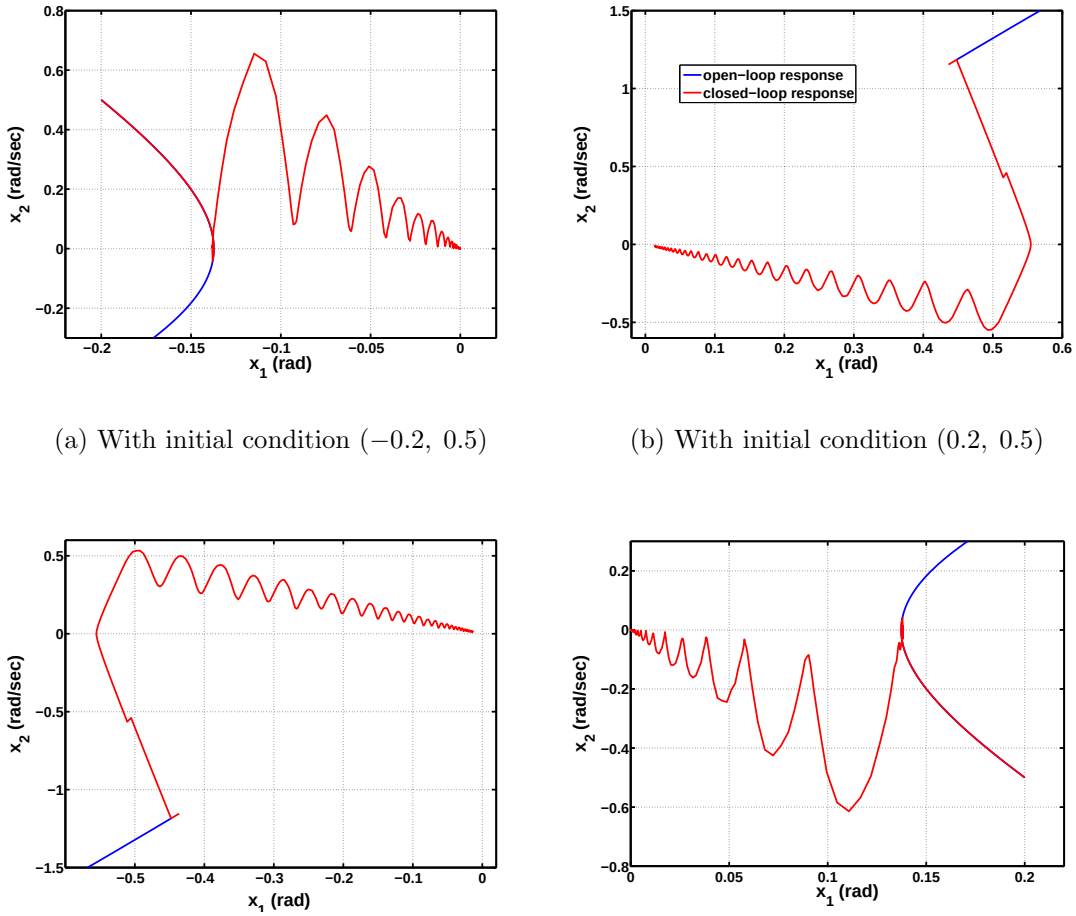
(c) With initial condition $(-2,\ -5)$

(d) With initial condition $(2,\ -5)$

Figure 4.6: Comparison of phase portraits with symmetrical initial conditions

$(0.2,\ 0.5)$ and $(0.2,\ -0.5)$, $(-0.2,\ 0.5)$ and $(-0.2,\ -0.5)$, the simulation results are still almost symmetrical, as shown in figure 4.7, which proves that the proposed control can stabilize the unknown pendulum system without dependence on the initial conditions.

From the first example it can be seen that the proposed cognition-oriented stabilization approach with RNN being the system identifier can produce symmetric closed-loop system responses with respect to symmetric initial conditions. Moreover, the scaling of different initial conditions does not influence that stabilization result. This fact shows that the correctness of the adaptivity of the proposed control with respect to different motions described by the same nonlinear system.



(a) With initial condition $(-0.2,\ 0.5)$

(b) With initial condition $(0.2,\ 0.5)$

(c) With initial condition $(-0.2,\ -0.5)$

(d) With initial condition $(0.2,\ -0.5)$

Figure 4.7: Comparison of phase portraits with symmetrical initial conditions changed in the same scaling

4.2 Example of stabilizing an unknown nonlinear aeroelastic wing system

4.2.1 Introduction to aeroelastic control of nonlinear lifting surfaces

It is well-known that instability in aeroelastic systems, such as aircraft flight vehicles, wind turbines, long span bridges, and skyscrapers, can cause Limit Cycle Oscillations(LCO), flutter, and even chaotic vibrations [DES03]. Under these cases significant decays of the flutter speed may happen and cause unexpected or even fatal accidents. Although one can suppress the aeroelastic instabilities by simply increasing the stiffness of the structure, in aeronautic context this passive strategy

would increase the structure weight and decrease the overall performance of the flight vehicles. As a result, implementation of active control techniques is necessary and meaningful to prevent the occurrence of aeroelastic instability, which is a mandatory requirement to maintain successful performance of aeroelastic wing systems.

In studies of flutter suppression of nonlinear systems, an aeroelastic model has been developed based on the research of the Benchmark Active Control Technology (BACT) wind-tunnel model designed at the NASA Langley Research Center [Was97], [SHWD00], [BSW00] and [Muk00]. For this kind of model a set of wind-tunnel tests have been performed to examine the effect of nonlinear structural stiffness.

To suppress the instability caused by these factors, different control systems have been designed in the past fifteen years using feedback linearizing technique [ZS98, XS00], model reference adaptive control approaches, back-stepping design methods [SW02], robust control design with proportional integral observer (PI-O) [ZS09], and so on. These methods stand for classical model-based approaches dealing with the effect of structural nonlinearities in aeroelastic problems.

However, it is difficult to establish a precise model of an aeroelastic system due to the variation of system dynamics and the strong nonlinearities caused by the coupling between the structural and aero dynamics. The methods mentioned above have their own limitations because of their fixed dynamical behaviors which cannot guarantee a required control performance especially in the case of unknown effects like modeling errors or unknown inputs acting on the system.

Compared to the aforementioned method, the cognition-oriented stabilization has inherently the superiority of adaptivity to the nonlinearities and unknown dynamics. In this section, the simulation results of the proposed cognition-oriented stabilization problem are presented to show the flexibility of the proposed control strategy.

4.2.2 Configuration of the aeroelastic system

The BACT wing-flap model has been widely studied in the aeroelastic research. The configuration of the nonlinear 2-D prototypical aeroelastic wing is shown in Fig.4.8. The two degrees of freedom, the pitching movement and the plunging one, are respectively restrained by a pair of springs attached to the elastic axis(EA) of the airfoil. A single trailing-edge control surface is used to control the air flow, thereby providing more maneuverability to suppress instability. This model is accurate for airfoils at low velocity and has been confirmed by wind tunnel experiments.

The equations of motion governing the aerolastic system are given as

$$\begin{aligned}\begin{bmatrix} m_T & m_W x_\alpha b \\ m_W x_\alpha b & I_\alpha \end{bmatrix} \begin{bmatrix} \ddot{h} \\ \ddot{\alpha} \end{bmatrix} + \begin{bmatrix} c_h & 0 \\ 0 & c_\alpha \end{bmatrix} \begin{bmatrix} \dot{h} \\ \dot{\alpha} \end{bmatrix} \\ + \begin{bmatrix} k_h & 0 \\ 0 & k_\alpha \end{bmatrix} \begin{bmatrix} h \\ \alpha \end{bmatrix} = \begin{bmatrix} -L \\ M \end{bmatrix},\end{aligned} \tag{4.4}$$

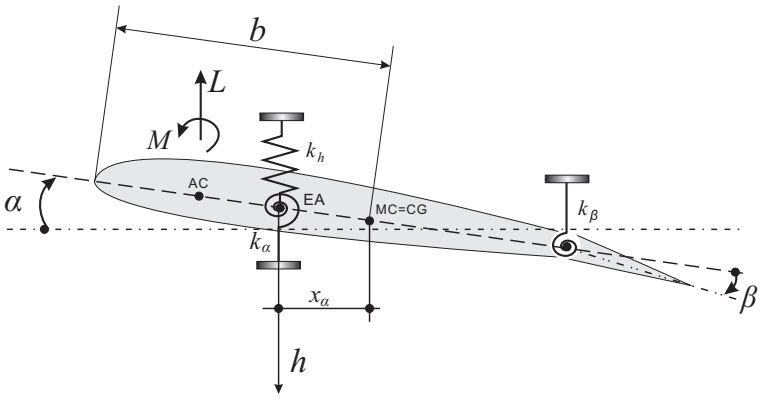


Figure 4.8: 2-D Wing-flap aeroelastic model.

where plunging and pitching displacement are denoted as h and α respectively. In Eq. (4.4) m_W denotes the mass of the wing, m_T represents the total mass of the wing and its support structure, b the semi-chord of the wing, I_α the moment of inertia, x_α the non-dimensional distance from the center of mass to the elastic axis, c_α and c_h the pitch and plunge damping coefficients respectively, k_α and k_h the pitch and plunge spring constants respectively, and M and L denote the quasi-steady aerodynamic lift and moment. In the case when the quasi-steady aerodynamics is considered, M and L should be written as

$$\begin{cases} L = \rho U^2 b c_{l_\alpha} \left[\alpha + \frac{\dot{h}}{U} + (\frac{1}{2} - a) b \frac{\dot{\alpha}}{U}\right] + \rho U^2 b c_{l_\beta} \beta \\ M = \rho U^2 b^2 c_{m_\alpha} \left[\alpha + \frac{\dot{h}}{U} + (\frac{1}{2} - a) b \frac{\dot{\alpha}}{U}\right] + \rho U^2 b^2 c_{m_\beta} \beta \end{cases}, \tag{4.5}$$

where the meaning of denotations can be found in tabular 4.2.2.

Table 4.1: Denotation list of aerodynamic coefficients

a	nondimensional distance from mid chord to elastic axis	c_h, c_α	pressure coefficients
b	semi-chord of the wing	U	free stream velocity
x_α	distance from elastic axis to mass center	ρ	density of air
$c_{m\alpha}$ c_{m_α}	the lift and moment coefficients per angle of attack	c_{l_β}, c_{m_β}	lift and moment coefficients per angle of control surface deflection

The control objective is to drive the flap angle β properly so that the instability caused by structural nonlinearities can be suppressed in the vicinity of the nominal system flutter speed with smaller control errors and less input energy. It is supposed that the displacement and the velocity of the pitching motion, α and $\dot{\alpha}$, can be

measured. The structural nonlinearity is supposed to exist in the pitching spring constant k_α and is assumed as to be a polynomial of α, shown as

$$k_\alpha = \sum_{i=0}^{4} k_{\alpha_i} \alpha^i. \tag{4.6}$$

4.2.3 Problem settings and control task

It is pointed in [LP66] that the pitching and plunging motion of the wing section of an aircraft or undersea craft is a symmetric, nonlinear, multivariable system; and for this kind of system, it is possible to utilize the system symmetry to bring facilitation to control. One of such possible facilitation is to use partitioned motion to control the global system, for example, the concerned aeroelastic system may be controlled by only controlling the pitching motion or plunging motion. Furthermore, in [SW02] it is proven that if the concerned system can be feedback linearized with respect to the pitching motion. Due to this knowledge, it is reasonable to assume in this example that only the pitching motion is fully measurable, i.e., the states α and $\dot{\alpha}$ are known. Other assumptions in this control task, which is similar to the assumptions in the pendulum examples, are listed as follows

- the measurements are noise-free;
- only the pitch motion α and $\dot{\alpha}$ are measured; and
- no information of the nonlinear dynamics is known.

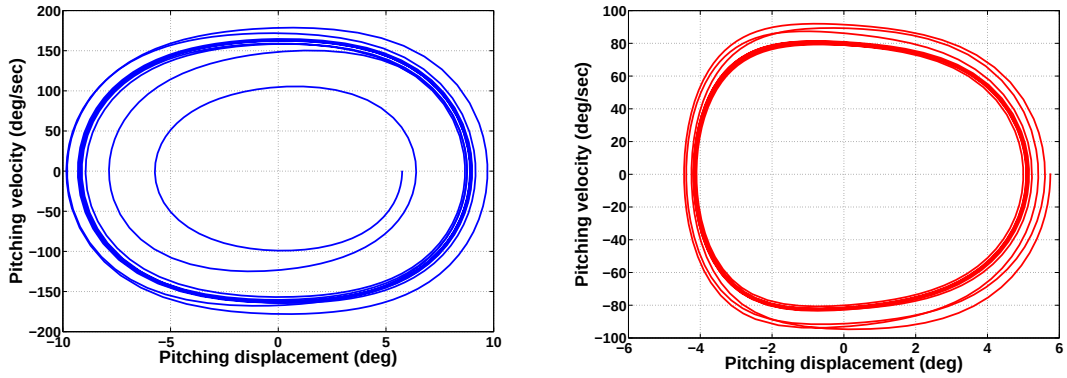
The task of control is defined as to stabilize the system following the above assumptions with two different nonlinearities of the pitching spring stiffness, as

$$k_{\alpha 1} = \left[\begin{array}{ccccc} 6.8 & 10.0 & 667.7 & 26.6 & -5087.9 \end{array} \right] [\alpha^i], \tag{4.7}$$

and

$$k_{\alpha 2} = \left[\begin{array}{ccccc} 2.8 & -62.3 & 3709.7 & -24195.6 & 48756.9 \end{array} \right] [\alpha^i]. \tag{4.8}$$

The data of the two nonlinearities are taken from [KSK99] and [ABG06], respectively. It should be mentioned that in order to examine the adaptive ability, the settings of the proposed controller should not be changed when the nonlinearity is changed into another one.



(a) Open-loop response with $k_{\alpha 1}$ (b) Open-loop response with $k_{\alpha 2}$

Figure 4.9: Open-loop system phase portraits with different nonlinearities

4.2.4 Simulation results

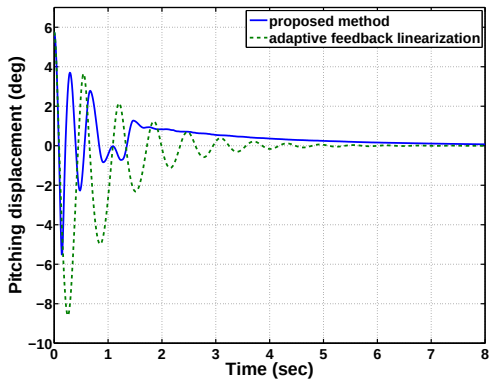
The values of the model parameters are taken from [ABG06] as

$$\begin{array}{lll} \rho = 1.225\,\mathrm{kg/m}^3 & b = 0.135\,\mathrm{m}\,, & c_{l_\alpha} = 6.28\,, \\ c_\alpha = 17.43\,\mathrm{Ns/m}\,, & c_h = 27.43\,\mathrm{Ns/m}\,, & c_{l_\beta} = 3.358\,, \\ k_h = 2844.4\,\mathrm{N/m}\,, & c_{m_\alpha} = (0.5+a)c_{l_\alpha}\,, & c_{m_\beta} = -0.635\,, \\ m_W = 2.0490\,\mathrm{kg}\,, & x_\alpha = [0.0873 - (b + a\,b)]/b\ \mathrm{m}\,, & \\ m_T = 12.387\,\mathrm{kg}\,, & \text{and } I_\alpha = m_W\,x_\alpha^2\,b^2 + 0.0517\,\mathrm{kg/m}^2\,. & \end{array}$$

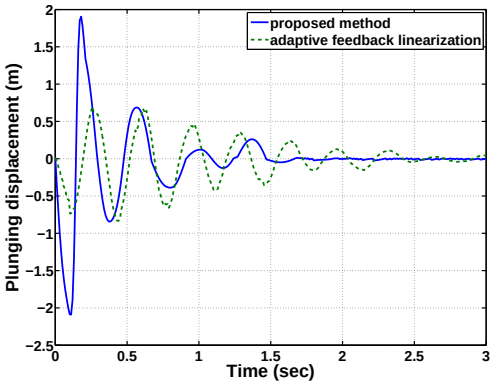
The open-loop responses of the concerned aeroelastic system with different nonlinearities $k_{\alpha 1}$ and $k_{\alpha 2}$ are shown in the figure 4.9(a) and the figure 4.9(b). It can be seen from these two figures that the structural nonlinearity in the pitching stiffness leads the system response to the Limit Cycle Oscillation (LCO).

In the closed-loop control of this example, the Radial Basis Function (RBF) neural network is utilized to learn and represent the plant dynamics, in contrast to the RNN network used in the introducing example. It has been proved that RBF neural network can approximate any kinds of continuous nonlinear functions within the neighborhood of present data [Pow89]. Furthermore, the RBF neural network is also suitable for incremental learning, which is also suitable for online applications. A detailed discussion on the training of the RBF neural network is omitted in this paper. Details can be found in [CCG91] and [SJ07]. The obtained RBF network model will be stored in the learned knowledge base for the use of planning.

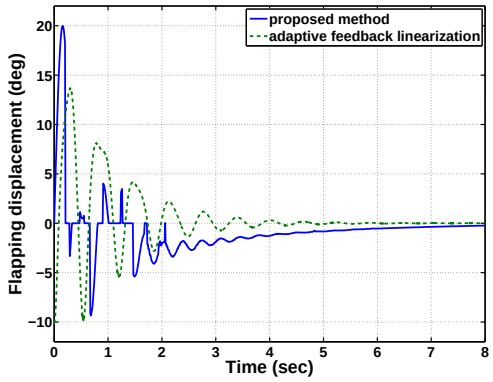
The system functionality measure in this example uses the form similar to the objective function in linear optimal control, i.e., the form introduced in equation (3.7)



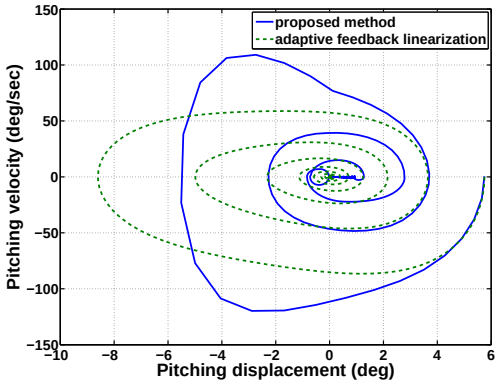
(a) Time history of pitching motion



(b) Time history of plunging motion



(c) Time history of control input



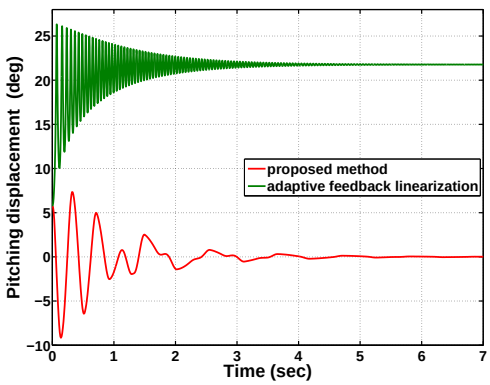
(d) Phase portrait of pitching motion with control

Figure 4.10: Closed-loop response with nonlinearity $k_{\alpha 1}$

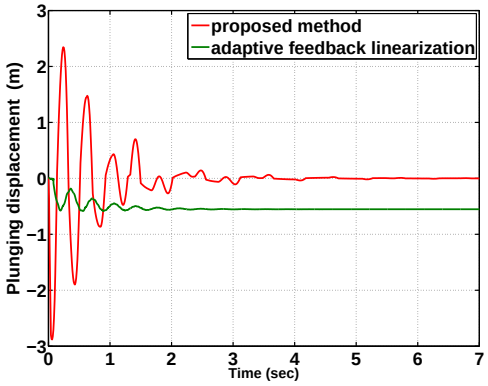
in chapter 3. The optimization problem (3.8) here is also solved by heuristic search. The searching space here is defined as the varying region of the flapping motion β, which is detailed as $(-45\text{deg},\ 45\text{deg})$ in the simulation.

Simulations of the close-loop system are performed with wind speed $U = 20\text{m/s}$ and structural parameter $a = 0.8$ (nondimensional distances from midchord to the elastic axis). The initial conditions for the state variables of the system are selected as $\alpha(0) = 5.75\,(\text{deg})$, $h(0) = 0.01\text{m}$, $\dot{\alpha}(0) = 0\,(\text{deg/s})$, and $\dot{h}(0) = 0\text{m/s}$. The sampling time is set as $1 \times 10^{-4}\text{sec}$.

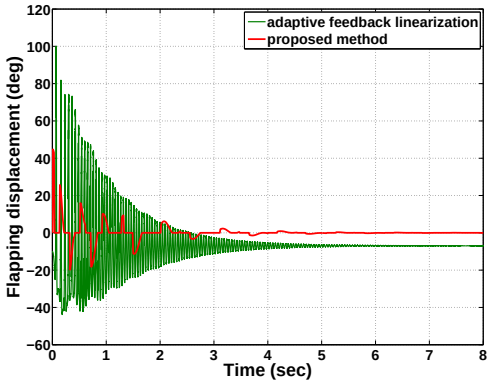
Furthermore, the adaptive feedback linearization control method, which is proposed in [KSK99] and designed for stabilization of the aeroelastic system with the first



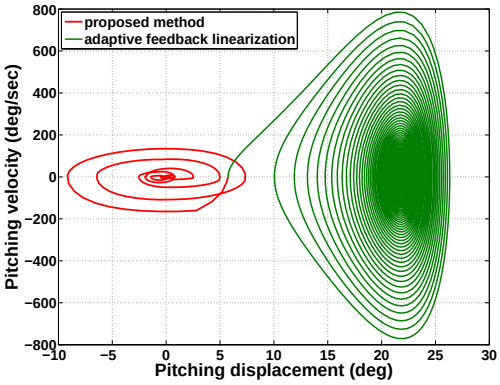
(a) Time history of pitching motion



(b) Time history of plunging motion



(c) Time history of control input



(d) Phase portrait of pitching motion with control

Figure 4.11: Closed-loop response with nonlinearity $k_{\alpha 2}$

nonlinearity $k_{\alpha 1}$, is also applied in the simulation as a comparison of the proposed control method. The setting of the adaptive feedback linearization controller used in this thesis is as the same as that used in [KSK99] and is not changed in the second simulation when the nonlinearity $k_{\alpha 1}$ is changed into $k_{\alpha 2}$.

Figure 4.10 and figure 4.11 show the simulation results of the closed-loop system with the nonlinearity $k_{\alpha 1}$ and $k_{\alpha 2}$, respectively. The blue curves in figure 4.10 and the red curves in figure 4.11 show the simulation results of the system under proposed control method, from which it can be seen that the both motions of the system converge to the origin of the state space with time going, which shows that the proposed control method can stabilize the unknown nonlinear aeroelastic system with different nonlinearities. This fact proves that the proposed control strategy

can not only be used to stabilize the unknown nonlinear system, but also adapt the variation of system dynamics and realize the cognition-oriented stabilization.

The simulation results of adaptive feedback linearization control with respect to the two different nonlinearities are shown as the green curves in figure 4.10 and figure 4.11, respectively. In comparison with the proposed cognition-oriented stabilization techniques, it can be seen that although for the system with the first nonlinearity $k_{\alpha 1}$ the adaptive feedback linearization can stabilize the system and has better control performance with respect to the error and settling time than the proposed method, it cannot stabilize the system to the desired position (origin in the state space) in the second simulation with $k_{\alpha 2}$, if its parameter settings in the second case are kept as the same as those in the first simulation. On the other hand, despite of the worse performance, the proposed method can stabilize the both cases without any intervention of human like changing control parameters and so on. From this comparison it is shown that the proposed method possesses better adaptivity with respect to the adaptive feedback linearization method.

4.3 Summary of this chapter

In this chapter, two numerical examples are given to examine the performance of the proposed control method. The first example is to keep a pendulum system quadratic stable at its inverted position. Recurrent-neural-network-based system identification method is utilized to learn the plant dynamics. The simulation results of the pendulum example produce symmetric closed-loop system responses with respect to symmetric initial conditions, which certificate the correctness of the adaptivity of the proposed control.

The second example is to stabilize a two-dimensional benchmark nonlinear aeroelastic system, where the radial-basis-function network is used to identify the plant dynamics. The simulations are run with two different structural nonlinearities and the results show that the proposed method can stabilize the target system with any of the two nonlinearities with the same controller settings. These results are compared with the simulation results of the system controlled by the adaptive feedback linearization method proposed in [KSK99]. Although the adaptive feedback linearization method can achieve successful stabilization results for the system with the first nonlinearity and owns better control performances than the proposed method, it fails in the stabilization of the system with the other nonlinearity. The comparison between the two types of controllers proves the independence of the proposed cognition-oriented stabilization method from the dynamics of the system to be controlled. Based on these successful simulation results and especially the adaptivity of the proposed control to different systems, it can be concluded that the proposed cognition-oriented stabilization approaches have cognitive capabilities in stabilization problems.

5 Summary and Outlook

5.1 Summary

The basis of this thesis to realize cognitive capabilities in automatic control systems lies in the leading point of view that cognition is in term of representational structures of the external world and the information process taking place based on these structures, which implies that not only biological systems, but also technical systems can be cognitive systems. The three-level hierarchy of cognitive control, which is developed under the understanding of cognition mentioned above, is briefly introduced in this thesis. From this perspective, a short review of the contemporary control field is made and shows that current control approaches are not cognitive because of the lack of the self-organization and active utilization of the learned knowledge.

The strategy of this contribution to endow control systems with cognitive abilities is similar to the existing strategy of realizing cognition in engineering applications: to integrate different basic cognitive functions and the expert knowledge suitable for one specific context into a suitable cognitive architecture. Following this strategy, the main task in this thesis is set as

- establishing a stability criterion which can be utilized by the controller itself to represent the abstract concept of stability,
- finding a cognitive architecture suitable for control problems, and
- integrating the stability criterion (as expert knowledge) and the soft-computing techniques (as basis cognitive functions) into the suitable cognitive architecture.

The core of this thesis discusses and details the expert knowledge used for cognition-oriented stabilization: a new stability judgment method for online data-driven quadratic stability judgment of unknown nonlinear discrete-time systems. It is shown in the second chapter that the existence of a quadratic Lyapunov function is identical to the existence of a suitable orthogonal matrix with which all the system states can be mapped into a negative halfspace. Using the connections between quadratic Lyapunov function and convex cones, the problem of stability assessment is converted into determining the emptiness of the intersection between the n-dimensional real positive space and the convex cone generated by the data set transformed with an orthogonal matrix from the measured systems states. This problem can be coped with by solving a max-min problem: by finding the orthogonal matrix which maximizes the minimum of the quadratic programming, the stability judgment can be given according to the sign of the optimized value of the max-min problem. The orthogonal matrices are constructed randomly according to a random

parametric representation which can cover the complete set of $n \times n$ orthogonal matrices. The max-min problem is solved with genetic algorithm because of the randomness of constructing orthogonal matrices. From the numerical examples representing different dynamical behaviors it can be seen that the stability judgments given by the proposed data-driven stability judgment method are consistent with the real stability behaviors of the concerned systems, showing the effectiveness of the proposed method.

Motivated by the basic idea to realize cognition-oriented stabilization, this contribution proposes a framework of cognition-oriented stabilization based on a cognitive architecture. The proposed stability criterion is utilized as the conceptual expert knowledge in the proposed framework, because the abstract concept of quadratic stability is represented geometrically and can be manipulated (numerically realizable) by the controller. Furthermore, this thesis reformulates the proposed stability criterion to indicate the feasible region of system responses which can lead to stable closed-loop dynamics, which is taken as the skillful expert knowledge within the framework and used by the planning module of the framework to generate situated control input.

Besides the expert knowledge, the proposed framework contains another five modules: perception, interpretation, learned knowledge, planning, and execution. Perception module prepares and structures the measured data for interpretation module; the interpretation module identifies the plant dynamics and judge stability with guidance of expert knowledge about stability; the identified plant dynamics and stability judgments are saved as learned knowledge; according to both the learned knowledge and the expert knowledge, planning module plans goal dynamics and situated control input functions; and execution module generates input values based on the generated input function and feeds the input back to the plant. By proper organization of the learning functions in the interpretation module and the utilization of the learned plant dynamics in the planning module, a cognition-oriented controller used for quadratic stabilization can be developed.

The proposed cognition-oriented stabilization method is applied into numerical examples to examine its performance. The first example is about the stabilization of a pendulum system with respect to the upper equilibrium point. Recurrent-neural-network-based system identification method is utilized as the learning function of plant dynamics in this. The simulation results of the pendulum example produce symmetric closed-loop system responses with respect to symmetric initial conditions, which certificate the correctness of the adaptivity of the proposed control.

The second example is to stabilize a two-dimensional benchmark nonlinear aeroelastic system. Unlike the introducing example, the interpretation module in the second example utilizes the radial-basis-function networks to identify the plant dynamics. To test the adaptivity of the proposed cognition-oriented stabilization method, the simulations are run with two different structural nonlinearities and compared with

the simulation of the established adaptive feedback linearization method. The simulation results show that without tuning controller structures or parameters, the proposed method can stabilize the target system with any of the two nonlinearities. On the other hand, the adaptive feedback linearization method can achieve successful stabilization results for the system with only one of the two nonlinearities. It cannot stabilize the system with the other nonlinearity with the same controller structure and parameter settings. The comparison between the two types of controllers proves the independence of the proposed cognition-oriented stabilization method from the dynamics of the system to be controlled. Based on these successful simulation results and especially the representation of the stability of the external world, it can be concluded that the proposed method possess cognitive capabilities for stabilization problems.

5.2 Limitations

There are still many limitations for the finished work, which provides directions to improve the proposed methods. These limitations exist mainly in three aspects and are stated as follows.

- The first limitation is the necessity of the stability judgment of the proposed online stability assessment algorithm. Although this limitation can be conditionally avoided, as discussed in section 2.6.2, it restricts the application area of this algorithm, especially in the data-driven stability analysis.
- The second limitation is about the conservativeness of the proposed stability criterion. The proposed criterion is restricted to quadratic stability, because the non-existence of a QLF does not support the fact that the system must be unstable, as there may exist other kinds of Lyapunov functions for the concerned system. This limitation may lead to the fact that a stable motion is judged as unstable by the proposed criterion.
- The third limitation is about the calculation power required by the data-driven stability judge. Although the calculation efficiency has been discussed in this thesis, the proposed method for data-driven stability judgment needs much more calculation powers when the system dimension increases, which is mainly caused by the random optimization techniques required by the stochastic construction of the full set of orthogonal matrices.
- The last limitation lies in the searching algorithm of suitable feedback control gains. Although this searching problem is expressed in this thesis as an optimization problem, it is not solved by optimization techniques but the heuristic search methods. It is well-known that heuristic search methods are usually slower than most of the real optimization approaches, when the optimization problem is well defined.

5.3 Future work

Correspondingly, the improvements of this work should be made in accordance with the limitations mentioned above.

The problem of the necessity may be solved by taking the data structure into consideration. Because the stability is judged based on the mapping by use of an orthogonal matrix and an entry-like matrix multiplication, shown as in equation (2.24), if the data space can be represented by some measured system states and the geometry of the mapped space can also be determined, the stability of the concerned system can be determined from the geometry of the mapped space. If this is possible, the problem of judging stability only for a single trajectory can be avoided, by which way the necessity problem can be solved.

For the conservativeness problem arise two possibilities to broaden this conservation. The first one is to introducing piece-wise quadratic Lyapunov function, similar to the study of common Lyapunov functions. The second may be to establish a generic description of positive definite functions and interpret this description in a geometrical manner, just as this thesis does. Both the two possibilities require deeper discussions of the geometry and the convex function analysis.

The third limitation can be improved by introducing another efficient parametric construction method of the complete set of orthogonal matrices and the corresponding optimization techniques. However, due to the fact that a n-dimensional orthogonal matrix must have $n(n-1)/2$ parameters, if the system order is too high, the labor for the computation would still be heavy. Therefore, in addition to introducing new methods of constructing orthogonal matrices and the relevant optimization method, a proper method to reduce the system dimension which can maintain the stability condition of the original system would be also helpful.

Similar to the third aspect of improvement, the last limitation can also be improved by introducing new optimization techniques. However, because the stability constraints provided by the proposed data-driven stability judgment is a complex nonlinear function, the feasibility of this optimization problem must be taken into consideration. A possible solution to guarantee the feasibility may to integrate this constraint into the searching mechanism in the optimization solver.

Besides the aforementioned possible improvements, another improving aspect may be about the proposed cognitive architecture. It is mentioned in the thesis that the Allocation of Resources module should perform the task to organize the topology of some cognitive functions, which is unfortunately in this contribution realized in fact by human understanding about stabilization problem, not by the architecture itself. If such functionalities can be realized, the cognitive architecture would own less dependence to human and may react actively and creatively to the context stimuli, rather than the regulated actions defined by the expert knowledge at the current stage.

Appendix

The parametric orthogonal matrices representation

According to [TL08], an orthogonal matrix can be denoted as

$$\phi = [\phi_1 \ \phi_2 \ ... \ \phi_N] \ ,$$

with

$$\begin{cases} \phi_1 = n_1(\theta_1) \\ \phi_i = (Q_1 Q_2 ... Q_i) n_i(\theta_i), \quad i =, 2 \, , ... \, N-2 \\ \phi_N = Q_1 Q_2 ... Q_{N-1} \end{cases} .$$

The definition of the parameters inside the above equation can be stated as follows:

- θ_i, a $(N-i) \times 1$ dimensional vector containing $N-i$ parameters

$$\theta_i = \begin{bmatrix} \theta_{i1} \\ \theta_{i2} \\ ... \\ \theta_{i(N-i)} \end{bmatrix}_{(N-i)\times 1} , \quad i = 1, \, ... \, , N-2.$$

- $n_i(\theta_i)$, a $(N-i+1) \times 1$ dimensional vector

$$n_i(\theta_i) = \begin{bmatrix} \sin\theta_{i1} \\ \cos\theta_{i1} \sin\theta_{i2} \\ ... \\ \left(\prod_{j=1}^{N-i-1} \cos\theta_{ij}\right) \sin\theta_{i(N-i)} \\ \prod_{j=1}^{N-i} \cos\theta_{ij} \end{bmatrix}_{(N-i+1)\times 1}$$

- Q_i, a $(N-i+1) \times (N-i)$ matrix, constructed by the following process:

  1. After $n_i(\theta_i)$ being obtained, construct *randomly* another $N-i-1$ vectors b_j, $j = 2$, ..., $N-i+1$,in such a way that $n_i(\theta_i)$ and these constructed vectors form a base of R^{N-i+1}, denoted as

  $$B_i = [n_i(\theta_i), b_2, \, ..., \, b_{N-i+1}]_{(N-i+1)\times(N-i+1)}$$

  2. Execute Gram−Schmidt process to the base B_i to obtain an orthonormal base of R^{N-i}. Let $g_1 = n_i(\theta_i)$, and do the following calculation

  $$g_{j+1} = b_{j+1} - \sum_{j=1}^{j} \frac{\langle b_{j+1}, g_j \rangle}{\langle g_j, g_j \rangle}, q_j = \frac{g_{j+1}}{\|g_{j+1}\|} \, ,$$

  for $j = 1, 2, ..., N-i$. Then the orthonormal base is obtained as $[n_i(\theta_i), \ q_1, ... \, q_{N-i}]$.

  3. The matrix Q_i can be obtained as

  $$Q_i = [\, q_1, ... \, q_{N-i}] \, .$$

Bibliography

[ABB+04] J. Anderson, D. Bothell, M. Byrne, S. Douglass, C. Lebiere, and Y. Qin, "An integrated theory of the mind," *Psychological Review*, vol. 111, no. 4, pp. 1036–1060, 2004.

[ABG06] V. M. R. A. Behal, P. Marzocca and A. Gnann, "Nonlinear adaptive model free control of an aeroelastic 2-d lifting surface," *Journal of Guidance, Control, and Dynamics*, vol. 29, no. 2, pp. 382–390, 2006.

[AF92] D. Avis and K. Fukuda, "A pivoting algorithm for convex hulls and vertex enumeration of arrangements and polyhedra," *Discrete Computing Geometry*, vol. 8, no. 3, pp. 295–313, 1992.

[AS68] R. Atkinson and R. Shiffrin, "Human memory: a proposed system and its control processes," in *Psychology of Learning and Motivation*, K. Spence and J. Spence, Eds. New York, USA: Academic Press, 1968, vol. 2, pp. 89–195.

[AS08] E. Ahle and D. Söffker, "Interaction of intelligent and autonomous systems - part ii: realization of cognitive technical systems," *Mathematical and Computer Modeling of Dynamical Systems*, vol. 14, no. 1, pp. 319–339, 2008.

[AZSAM10] H. Abootorabi Zarchi, J. Soltani, and G. Arab Markadeh, "Adaptive input-output feedback-linearization-based torque control of synchronous reluctance motor without mechanical sensor," *IEEE Transactions on Industrial Electronics*, vol. 57, no. 1, pp. 375–384, 2010.

[Bar85] B. R. Barmish, "Necessary and sufficient conditions for quadratic stabilizability of an uncertain system," *Journal of Optimization Theory and Applications*, vol. 46, no. 4, pp. 399–408, 1985.

[BBG+11] S. Biundo, P. Bercher, T. Geier, F. Müller, and B. Schattenberg, "Advanced user assistance based on AI planning," *Cognitive Systems Research*, vol. 12, no. 3-4, pp. 219–236, 2011.

[BBW07a] M. Beetz, M. Buss, and D. Wollherr, "Cognitive technical systems - what is the role of artificial intelligence?" in *KI 2007: Advances in Artificial Intelligence*, ser. Lecture Notes in Computer Science, J. Hertzberg, M. Beetz, and R. Englert, Eds. Berlin, Germany: Springer, 2007, vol. 4667, pp. 19–42.

[BBW07b] M. Buss, M. Beetz, and D. Wollherr, "CoTeSys - cognition for technical systems," in *Proceedings of the 4th COE Workshop on Human Adaptive Mechatronics, HAM 2007*, Tokyo, Japan, March 2–3 2007.

[Ben79] S. Bennet, *A History of Control Engineering 1800-1930*. London, UK: Peter Peregrinus Ltd., 1979.

[BLL04] D. P. Benjamin, D. Lyons, and D. Lonsdale, "ADAPT: A cognitive architecture for robotics," in *Proceedings of the 2004 International Conference on Cognitive Modelling, ICCM 2004*, A. R. Hanson and E. M. Riseman, Eds., Pittsburgh, USA, July 30–August 1 2004, pp. 337–343.

[Bod00] M. A. Boden, "Autopoiesis and life," *Cognitive Science Quarterly*, vol. 1, no. 1, pp. 117–145, 2000.

[BSW00] R. M. Bennett, R. C. Scott, and C. Wieseman, "Computational test cases for the benchmark active controls model," *Journal of Guidance, Control, and Dynamics*, vol. 23, no. 5, pp. 922–929, 2000.

[BV04] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge, UK: Cambridge University Press, 2004.

[BW06] G. Bretthauer and D. Westerkamp, "Automation technology 2010: challenges and chances - nine theses concerning future development," *Automatisierungstechnik*, vol. 54, no. 9, pp. 459–461, 2006.

[BWB09] T. Batz, K. Watson, and J. Beyerer, "Recognition of dangerous situations within a cooperative group of vehicles," in *Proceedings of 2009 IEEE Intelligent Vehicles Symposium*, Xi'an, China, June 2–5 2009, pp. 907 –912.

[Cac98] P. Cacciabue, *Modeling and Simulation of Human Behaviour in System Control*. London: Springer, 1998.

[Car11] C. Carathéodory, "Über den variabilitätsbereich der fourierschen konstanten von positiven harmonischen funktionen," *Rendiconti del Circolo Matematico di Palermo*, vol. 32, no. 1, pp. 193–217, 1911.

[CCdF99] G. Chen, G. R. Chen, and R. de Figueiredo, "Feedback control of unknown chaotic dynamical systems based on time-series data," *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, vol. 46, no. 5, pp. 640–644, 1999.

[CCG91] S. Chen, C. F. N. Cowan, and P. M. Grant, "Orthogonal least squares learning algorithm for radial basis function networks," *IEEE Transaction on Neural Networks*, vol. 2, no. 2, pp. 302–309, 1991.

[CGH03] D. Cheng, L. Guo, and J. Huang, "On quadratic lyapunov functions," *IEEE Transactions on Automatic Control*, vol. 48, no. 5, pp. 885–890, 2003.

[CHI01] A. J. Calise, N. Hovakimyan, and M. Idan, "Adaptive output feedback control of nonlinear systems using neural networks," *Automatica*, vol. 37, no. 8, pp. 1201–1211, 2001.

[CLRS90] T. Cormen, C. Leiserson, R. Rivest, and C. Stein, "Finding the convex hull," in *Introduction to Algorithms*, 2nd ed. Cambridge, USA: MIT Press and McGraw-Hill, 1990, pp. 955–966.

[CTN07] H. Chong, A. Tan, and G. Ng, "Integrated cognitive architectures: a survey," *Artificial Intelligence Review*, vol. 28, no. 2, pp. 103–130, 2007.

[CZH09] A. Cao, B. Zhou, and C. Hou, "The model-free direct adaptive generalized predictive control approach of permanent magnet linear motor," in *Proceedings of the 2009 International Conference on Electrical Machines and Systems, ICEMS 2009*, Tokyo, Japan, November 15-18 2009, pp. 1–5.

[dBK99] F. de Bruyne and L. Kammer, "Iterative feedback tuning with guaranteed stability," in *Proceedings of the 1999 American Control Conference, ACC 1999*, San Diego, USA, June 2-4 1999, pp. 3317–3321.

[DBPL00] R. Decarlo, M. Branicky, S. Pettersson, and B. Lennartson, "Perspectives and results on the stability and stabilizability of hybrid systems," *Proceedings of the IEEE*, vol. 88, no. 7, pp. 1069–1082, 2000.

[DES03] E. Dowell, J. Edwards, and T. Strganac, "Nonlinear aeroelasticity," *Journal of Aircraft*, vol. 40, no. 5, pp. 857–874, 2003.

[DL05] H. Deng and H.-X. Li, "A novel neural approximate inverse control for unknown nonlinear discrete dynamical systems," *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 35, no. 1, pp. 115 –123, 2005.

[DlSAQ07] M. De la Sen and S. Alonso-Quesada, "Model matching via multirate sampling with fast sampled input guaranteeing the stability of the plant zeros: extensions to adaptive control," *IET Control Theory Applications*, vol. 1, no. 1, pp. 210–225, 2007.

[DO01] Y. Dote and S. Ovaska, "Industrial applications of soft computing: a review," *Proceedings of the IEEE*, vol. 89, no. 9, pp. 1243–1265, 2001.

[FBD+04] W. T. Fu, D. Bothell, S. Douglass, C. Haimson, M.-H. Sohn, and J. R. Anderson, "Learning from real-time over-the-shoulder instructions in a dynamic task," in *Proceedings of the 6th International Conference on Cognitive Modeling, ICCM 2004*, Pittsburg, USA, July 30–August 1 2004, pp. 100–105.

[FCZ03] E. Fernández-Cara and E. Zuazua, "Control theory: history, mathematical achievements and perspectives," *Boletín SEMA (Sociedad Española de Matemática Aplicada)*, vol. 26, no. 12, pp. 79–140, 2003.

[FJSR08] M. Fliess, C. Join, and H. Sira-Ramirez, "Nonlinear estimation is easy," *International Journal of Modeling Identification and Control*, vol. 4, no. 1, pp. 12–27, 2008.

[Fod09] J. Fodor, *Aspects of Soft Computing, Intelligent Robotics and Control*. Berlin, Germany: Springer, 2009.

[FP10] F. Farid and F. Pourboghrat, "Stabilizing control of a class of unknown nonlinear systems using dynamic neural networks," in *Proceedings of the 2010 American Control Conference, ACC 2010*, Baltimore, USA, June 30–July 2 2010, pp. 4919–4924.

[FS01] D. Foley and N. Sadegh, "Modelling of nonlinear systems from input-output data for state space realization," in *Proceedings of the 40th IEEE Conference on Decision and Control, CDC 2001*, Orlando, USA, December 4–7 2001, pp. 2980–2985.

[FS10] X. G. Fu and D. Söffker, "Cognitive awareness of intelligent vehicles," in *Proceedings of SAE World Congress of Intelligent Vehicle Initiative (IVI), Technology Advanced Control and Navigation*, Detroit, USA, April 13–15 2010.

[GS10] D. Gamrad and D. Söffker, "Representation of knowledge for technical systems to realize cognitive functions for mobile robotic applications," in *Entwurf komplexer Automatisierungssysteme - 11. Fachtagung mit Tutorium und Toolausstellung, EKA 2010*, Magdeburg, Germany, May 25–27 2010, pp. 277–286.

[Hah67] W. Hahn, *Stability of Motion*. New York: Springer, 1967.

[Hay99] S. Haykin, *Neural Networks*, 2nd ed. New Jersey, USA: Prentice Hall International, 1999.

[Hja05] H. Hjalmarsson, "From experiment design to closed-loop control," *Automatica*, vol. 41, no. 41, pp. 393–438, 2005.

[Hol75] J. H. Holland, *Adaption in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control and Artificial Intelligence*. Ann Arbor, USA: University Michigan Press, 1975.

[Hol77] E. Hollnagel, "Cognitive functions in decision making," in *Decision Making and Change in Human Affairs*, H. Jungermann and G. de Zeeuw, Eds. Amsterdam, the Netherland: D. Reidel, 1977.

[Ise98] R. Isermann, "On fuzzy logic applications for automatic control, supervision, and fault diagnosis," *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*, vol. 28, no. 2, pp. 221–235, 1998.

[JVL00] S. Jagannathan, M. W. Vandegrift, and F. L. Lewis, "Adaptive fuzzy logic control of discrete-time dynamical systems," *Automatica*, vol. 36, no. 2, pp. 229–241, 2000.

[JW71] R. Jonnada and C. Weygandt, "Limit cycles of higher order nonlinear autonomous systems," *Journal of the Franklin Institute*, vol. 291, no. 3, pp. 195–210, 1971.

[Kha02] H. K. Khalil, *Nonlinear Systems*, 3rd ed. New Jersey: Prentice Hall, 2002.

[Kos10] E. B. Kosmatopoulos, "Control of unknown nonlinear systems with efficient transient performance using concurrent exploitation and exploration," *IEEE Transactions on Neural Networks*, vol. 21, no. 8, pp. 1245–1261, 2010.

[KPKP05] J. Kim, C. Park, E. Kim, and M. Park, "Adaptive synchronization of T-S fuzzy chaotic systems with unknown parameters," *Chaos Solitons & Fractals*, vol. 24, no. 5, pp. 1353–1361, 2005.

[KSK99] J. Ko, T. W. Strganac, and A. J. Kurdila, "Adaptive feedback linearization for the control of a typical wing section with structural nonlinearity," *Nonlinear Dynamics*, vol. 18, no. 3, pp. 289–301, 1999.

[KWM97] D. E. Kieras, S. D. Wood, and D. E. Meyer, "Predictive engineering models based on the epic architecture for a multimodal high-performance human-computer interaction task," *ACM Transactions on Computer-Human Interaction*, vol. 4, no. 3, pp. 230–275, 1997.

[Lan95] P. Langley, "Order effects in incremental learning," in *Learning in Humans and Machines: Towards an Interdisciplinary Learning Science*, P. Reimann and H. Spada, Eds. Oxford, UK: Pergamon Press, 1995, pp. 1–17.

[Lan05] ——, "An adaptive architecture for physical agents," in *Proceedings of the 2005 IEEE/WIC/ACM International Conference on Intelligent Agent Technology, IEEE/WIC/ACM-IAT 2005*, Compiegne, France, September 19–22 2005, pp. 18–25.

[Lan06] ——. (2006, October) A cognitive architecture for physical agents. [Online]. Available: http://www.isle.org/langley/talks/icarus.6.04.ppt

[LC06] P. Langley and D. Choi, "A unified cognitive architcture for physical agents," in *Proceedings of the 21st National Conference on Artificial Intelligence, AAAI-06*, N. Smelser and P. Baltes, Eds., Boston, USA, July 16–20 2006, pp. 1469–1474.

[Len08] G. G. Lendaris, "Higher level application of adp: a next phase for the control field?" *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, vol. 38, no. 4, pp. 901–912, 2008.

[Lil10] J. H. Lilly, *Fuzzy Control and Indentification*. New Jersey, USA: John Wiley & Sons, 2010.

[Liu01] G. P. Liu, *Nonlinear Identification and Control: A Neural Network Approach*. London, UK: Springer, 2001.

[LLR98] J. F. Lehman, J. Laird, and P. Rosenbloom, "A gentle introduction to Soar: an architecture for human cognition," in *Invitation to Cognitive Science*, S. Sternberg and D. Scarborough, Eds. Cambridge, USA: MIT Press, 1998.

[LLR09] P. Langley, J. E. Laird, and S. Rogers, "Cognitive architectures: Research issues and challenges," *Cognitive Systems Research*, vol. 10, no. 2, pp. 141–160, 2009.

[Lov01] M. C. Lovett, "Cognitive theory: Act," in *International Encyclopedia of Social and Behaviorial Sciences*, N. Smelser and P. Baltes, Eds. Amsterdam, the Netherland: Elsevier, 2001, vol. 3, pp. 2175–2178.

[LP66] A. G. Lindgren and R. F. Pinkos, "Stability of symmetric nonlinear multivariable systems," *Journal of the Franklin Institute*, vol. 282, no. 2, pp. 92–101, 1966.

[LQZK04] Y. H. Li, S. Q., X. Y. Zhuang, and O. Kaynak, "Robust and adaptive backstepping control for nonlinear systems using rbf neural networks," *IEEE Transactions on Neural Networks*, vol. 15, no. 3, pp. 693–701, 2004.

[Lya92] A. Lyapunov, *The General Problem of the Stability of Motion*, A. Fuller, Ed. Bristol: Taylor & Francis, 1992.

[MB96] D. McFarland and T. Bösser, *Intelligent Behavior in Animals and Robots*. Cambridge, USA: MIT Press, 1996.

[MGP60] G. A. Miller, E. Galanter, and K. H. Pribram, *Plans and the Structure of Behavior*. New York, USA: Holt, Rinehart & Winston, 1960.

[Mil79] G. A. Miller, *A Very Personal History (Occasional Paper No. 1)*. Cambridge, USA: Center for Cognitive Science, 1979.

[MR08] I. Markovsky and P. Rapisarda, "Data-driven simulation and control," *International Journal of Control*, vol. 81, no. 6, pp. 1946–1959, 2008.

[Muk00] V. Mukhopadhyay, "Transonic flutter suppression control law design and wind-tunnel test results," *Journal of Guidance, Control, and Dynamics*, vol. 23, no. 5, pp. 930–937, 2000.

[Nei76] U. Neisser, *Cognition and Reality: Principles and Implications of Cognitive Psychology*. New York: W.H. Freeman and Company, 1976.

[New90] A. Newell, *Unified Theories of Cognition*. Cambridge, USA: Harvard University Press, 1990.

[Nor96] D. A. Norman, "Cognitive engineering," in *New Thinking in Design: Conversations on Theory and Practice*, G. T. Mitchell, Ed. New York, USA: Van Nostrand Reinhold, 1996, pp. 87–99.

[OHS10] H. Oberheid, A. Hasselberg, and D. Söffker, "Know your options – analyzing human decision making in dynamic task environments with state-space methods," in *Proceedings of Human Factors and Ergonomics Society – Annual Meeting of the Europe Chapter, HFES 2010*, Berlin, Germany, October 13–16 2010.

[PI09] U. S. Park and M. Ikeda, "Stability analysis and control design of LTI discrete-time systems by the direct use of time series data," *Automatica*, vol. 45, no. 2, pp. 1265–1271, 2009.

[Pow89] M. J. D. Powell, "Radial basis function approximation to polynomials," in *Numerical Analysis 1987*. White Plains, New York: Longman Publishing Group, 1989, pp. 223–241.

[Rig09] G. G. Rigatos, "Adaptive fuzzy control of dc motors using state and output feedback," *Electric Power Systems Research*, vol. 79, no. 11, pp. 1579–1592, 2009.

[RJ00] J. Rickel and W. Johnson, "Task-oriented collaboration with embodied agents in virtual worlds," in *Embodied Conversational Agents*, J. Cassell, J. Sullivan, and S. Prevost, Eds. Boston: MIT Press, 2000, pp. 95–122.

[Rov99] G. Rovithakis, "Robust neural adaptive stabilization of unknown systems with measurement noise," *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 29, no. 3, pp. 453 –459, 1999.

[SA07] D. Söffker and E. Ahle, "Idea, conception, and realisation of learning abilities for robot control using a situation-operator-model," *International Journal of Intelligent Systems Technologies and Applications*, vol. 2, no. 2–3, pp. 271–283, 2007.

[Sav05] S. Savulescu, *Real-Time Stability in Power Systems*. New York: Springer, 2005.

[Sea90] J. R. Searle, "Is the brain a digital computer?" *Proceedings and Addresses of the American Philosophical Association*, vol. 64, no. 3, pp. 21–37, 1990.

[SG08] C. Stiller and J. Gayko, "Kognitive Automobile," *at - Automatisierungstechnik*, vol. 56, no. 11, pp. 551–553, 2008.

[SHH+95] G. Strube, C. Habel, B. Hemforth, L. Konieczny, and B. Becker, "Kognition," in *Einführung in die künstliche Intelligenz*, 2nd ed., G. Görz, Ed. Bonn, Germany: Addison-Wesley, 1995.

[SHWD00] R. C. Scott, S. T. Hoadley, C. D. Wieseman, and M. H. Durham, "The benchmark active controls technology model aerodynamic data," *Journal of Guidance, Control, and Dynamics*, vol. 23, no. 5, pp. 914–921, 2000.

[SJ07] K. Salahshoor and M. R. Jafari, "Online identification of nonlinear systems using adaptive rbf-based neural networks," *International Journal of Information Science and Technology*, vol. 5, no. 2, pp. 99–121, 2007.

[Sto73] S. Storøy, "An algorithm for finding a vector in the intersection of open convex polyhedral cones," *BIT Numerical Mathematics*, vol. 16, no. 1, pp. 459–461, 1973.

[Str96] G. Strube, "Kognition," in *Wörterbuch der Kognitionswissenschaft*, G. Strube, Ed. Stuttgard, Germany: Klett-Cotta, 1996, pp. 303–317.

[Str98] ——, "Modelling motivation and action control in cognitive systems," in *Mind Modeling*, U. Schmid, J. F. Krems, and F. Wysocki, Eds. Berlin, Germany: Pabst, 1998, pp. 111–130.

[Str01] ——, "Cognitive science: overview," in *International Encyclopedia of Social and Behaviorial Sciences*, N. Smelser and P. Baltes, Eds. Amsterdam, the Netherland: Elsevier, 2001, vol. 3, pp. 2158–2166.

[SW02] S. Singh and L. Wang, "Output feedback form and adaptive control of a nonlinear aeroelastic system," *Journal of Guidance, Control, and Dynamics*, vol. 25, no. 4, pp. 725–732, 2002.

[SZL+95] J. Sjoberg, Q. H. Zhang, L. Ljung, A. Benveniste, B. Delyon, P. Y. Glorennec, H. Hjalmarsson, and A. Juditsky, "Nonlinear black-box modeling in system identification: a unified overview," *Automatica*, vol. 31, no. 12, pp. 1691–1724, 1995.

[TC98] S. C. Tong and T. Y. Chai, "Fuzzy indirect adaptive control for a class of decentralized nonlinear systems," *International Journal of Systems Science*, vol. 29, no. 2, pp. 149–157, 1998.

[Tha10] P. Thagard. (2010, June) The stanford encyclopedia of philosophy: cognitive science. [Online]. Available: http://plato.stanford.edu/archives/sum2010/entries/cognitive-science/

[TL08] Y. Tang and J. P. Li, "A new algorithm of ICA: using the parametrized orthogonal matrixes of any dimensions," *Acta Automatica Sinica*, vol. 34, no. 1, pp. 31–39, 2008.

[Tul72] E. Tulving, "Episodic and semantic memory," in *Organization of Memory*, E. Tulving and W. Donaldson, Eds. New York, USA: Academic Press, 1972.

[UBL06] H. Ulbrich, T. Buschmann, and S. Lohmeier, "Development of the humanoid robot lola," *Journal of Applied Mechanics and Materials*, vol. 5-6, no. 5, pp. 529–540, 2006.

[vHdJS06] J. van Helvoort, B. de Jager, and M. Steinbuch, "Ellipsoidal unfalsified control: stability," in *Proceedings of the 2006 American Control Conference, ACC 2006*, Minneapolis, USA, June 14–16 2006, pp. 4094–4099.

[Was97] M. R. Waszak, "Robust multivariable flutter suppression for the benchmark active control technology (BACT) wind-tunnel model," *Journal of Guidance, Control, and Dynamics*, vol. 24, no. 1, pp. 143–147, 1997.

[WCL08] W.-Y. Wang, Y.-H. Chien, and I.-H. Li, "An on-line robust and adaptive T-S fuzzy-neural controller for more general unknown systems," *International Journal of Fuzzy Systems*, vol. 10, no. 1, pp. 33–43, 2008.

[WOW08] J. Wang, M. Obeng, and X. Wu, "Adaptive nn stabilization of uncertain nonholonomic mechanical systems," in *Proceedings of 2nd International Symposium on Systems and Control in Aerospace and Astronautics, ISSCAA 2008*, December 10–12 2008, pp. 1–6.

[WZ89] R. Williams and D. Zipser, "A learning algorithm for continually running fully recurrent neural networks," *Neural Computation*, vol. 1, no. 2, pp. 270–280, 1989.

[XS00] W. Xing and S. Singh, "Adaptive output feedback control of a nonlinear aeroelastic structure," *Journal of Guidance, Control and Dynamics*, vol. 23, no. 6, pp. 1109–1116, 2000.

[Zad94] L. A. Zadeh, "Fuzzy logic, neural networks, and soft computing," *Communications of the ACM*, vol. 37, no. 3, pp. 77–84, 1994.

[ZGH00] T. Zhang, S. S. Ge, and C. C. Hang, "Adaptive neural network control for strict-feedback nonlinear systems using backstepping design," *Automatica*, vol. 36, no. 12, pp. 1835–1846, 2000.

[Zie97] G. Ziegler, *Lectures on Polytopes*. New York, USA: Springer-Verlag, 1997.

[ZS98] Y. Zeng and S. Singh, "Output feedback variable structure adaptive control of an aeroelastic systems," *Journal of Guidance, Control, and Dynamics*, vol. 21, no. 6, pp. 830–837, 1998.

[ZS09] F. Zhang and D. Söffker, "Active flutter suppression of a nonlinear aeroelastic system using pi-observer," in *Motion and Vibration Control*, H. Ulbrich and L. Ginzinger, Eds. Dordrecht, The Netherlands: Springer, 2009, pp. 367–376.