INTERNATIONAL JOURNAL

of

COMPUTERS, COMMUNICATIONS & CONTROL

With Emphasis on the Integration of Three Technologies

Year: 2007 Volume: II Number: 4 (December)

CCC Publications

Licensed partner: EBSCO Publishing

www.journal.univagora.ro

ISSN 1841-9836 (print version), 1841-9844 (online version)

International Journal of Computers, Communications and Control (IJCCC) is published from 2006 and has 4 issues/year (March, June, September, December), print & online.

Founders of IJCCC: I. Dziţac, F.G. Filip and M.J. Manolescu (2006)

This publication is subsidized by:

1. Agora University
2. The Romanian Ministry of Education and Research / The National Authority for Scientific Research

CCC Publications, powered by Agora University Publishing House, currently publishes the "International Journal of Computers, Communications & Control" and its scope is to publish scientific literature (journals, books, monographs and conference proceedings) in the field of Computers, Communications and Control.

IJCCC is indexed and abstracted in a number of databases and services including:

1. Directory of Open Access Journals;
2. The Collection of Computer Science Bibliographies;
3. Open J-Gate;
4. FIZ KARLSRUHE's informatics portal;
5. EBSCO;
6. The Index of Information Systems Journals;
7. Urlich's Periodicals Directory,
8. SCIRUS;
9. e-Math for Africa;
10. Google Scholar, Google Academic.

Contents

International Journal of Computers, Communications & Control
Vol. II (2007), No. 4, pp. 303-313

DOMINO: Trivalent Logic Semantics in Bivalent Syntax Clothes

Boldur E. Bărbat

Abstract: The paper describes a rather general software mechanism developed primarily for decision making in dynamic and uncertain environments (typical application: managing overbooking). DOMINO (*D*ecision-*O*riented *M*echanism for "*I*F" as *N*on-deterministic *O*perator) is meant to deal with undecidability due to any kind of future contingents. Its description here is self-contained but, since a validation is underway within a much broader undertaking involving agent-oriented software, to impair redundancy, several aspects explained in very recent papers are here abridged. In essence, DOMINO acts as an "IF" with enhanced semantics: it can answer "YES", "NO" or "UNDECIDABLE in the time span given" (it renders control to an exception handler). Despite its trivalent logic semantics, it respects the rigours of structural programming and the syntax of bivalent logic (it is programmed in plain C++ to be applicable to legacy systems too). As for most novel approaches, expectations are high, involving a less algorithmic, less probabilistic, less difficult to understand method to treat undecidability in dynamic and uncertain environments, where *postponing decisions* means *keeping open alternatives* (to react better to rapid environment changes).
Keywords: undecidability; open, heterogeneous, dynamic and uncertain environments (OHDUE); decision-making; trivalent logic semantics; agent-oriented software engineering.

1 Introduction

Despite the major changes taking place in the Internet and globalization era - and their increasing speed because of the geometrically increasing computing power (due to Moore's law, that is expected to hold at least other ten years) -, basic software mechanisms advanced much too slowly. As regards uncertain knowledge processing, the hindrance is obvious: available software tools are either hardly affordable (because of high complexity - both cognitive and structural) or rather ineffective (designed for other environments, applied to ill-defined problems or lacking expected functionality). For instance, the very concept of "uncertainty" was treated inadequately - regardless of its growing relevance for application domains (mainly where real-time decision-making is involved [14]), environments (even more dynamic and uncertain [19]), end-user expectations (requesting anthropocentric interfaces [4]), IT paradigms (predominantly "computing as interaction" [1]) and so on. The main weaknesses: a) insufficient theoretical rigour (undecidability is considered primarily atemporal - keeping its initial mathematical meaning); b) poor practical effectiveness (confusing "unknown" with "unknowable" and applying sophisticated prediction methods in inappropriate contexts); c) unfit apparatus (algorithmic approaches implying determinism, decidability, and bivalence). Since the first two issues are investigated in very recent papers [8] [6] [9] [7], to reduce redundancy and to keep focusing on the very mechanisms, here the approach is an explicitly software engineering one: computer science aspects are dealt with only at the beginning (what for new mechanisms?) and at the end (what are the expectations?). In particular, the paper presents in detail DOMINO, a mechanism developed primarily for decision making in dynamic and uncertain environments (a typical example for potential application area comes from the overbooking policy of carrier companies). The *D*ecision-*O*riented *M*echanism for "*I*F" as *N*on-deterministic *O*perator (DOMINO) is meant to deal with undecidability due to any kind of future contingents. As a rather general software mechanism, its description here is self-contained. However, since a validation is underway within a much broader undertaking involving agent-oriented software (including non-algorithmic approaches to treat uncertainty), several aspects explained in the papers mentioned above are here abridged or skipped

over. As a result, the paper is structured as follows: Section 2 shows the *rationale* (i.e., *premises and diagnosis*) for developing new methods to handle uncertainty in decision support systems (DSS). Section 3 *delimits the problem* in both meanings: firstly it restricts it (narrowing the scope to undecidability due to future contingents) and secondly it *defines* it (as software engineering task). Section 4 outlines the architecture, basically *in search of the third value semantics*. On this groundwork, Section 5 explains the structure, in search of bivalent syntax clothes. Since conclusions are prohibited for a bottom-up project, before its validation, Section 6 lists the expectations, ranged in three time horizons (in fact, they are first conclusions).

2 Rationale. Premises and Diagnosis

Since in this section the paper is to some extent also a position paper, for the sake of simplicity, the premises, opinions, criteria, motives, and their corollaries are not separated in conceptual categories but asserted clustered around two "attractor words": *premises and diagnosis*. Thus, consensus is not mandatory to assess the research results. In other words, the utility of DOMINO can be evaluated even when some of the claims made here are rejected. Most of the assertions below are valid in any modern IT context but DSS are explicitly referred to because "IF" is the basic tool (again in both senses: *essential* and *simple*) for decision-making. Likewise, the emphasis is on the main changes affecting the essence of "IF". To preserve both text autonomy and shortness, for topics dealt with recently, details should be found in the paper the quotation comes from.

Premises. They reflect the general context, representing a very simplified input vector:

- *Environment*. "Present-day IT environments, except for some irrelevant applications, are open and heterogeneous (the resources involved are unalike and their availability is not warranted), dynamic (the pace of exogenous and endogenous changes is high) and uncertain (both information and its processing rules are revisable, fuzzy, uncertain and intrinsically non-deterministic" [6].

- *System*. Except trivial applications, the system exposed to decision-making is "man-machine system [...], highly complex (multi-distributed - mainly in space, but also in time and organization), under real-time control (the subsystems act on each other - at least, communicate intensely), almost always online" [7].

- *User expectations*. "Since most decision makers are already familiar with Google, most available general information is either known or easy accessed; now they need acceptable good answer, but very fast and with incomplete or even uncertain information" [7]. "Intelligent system behaviour - whatever that could mean - becomes a crucial user expectation" [8].

- *Task*. Most aim at "managing situations"; main attributes are: "high complexity, multicriterial, Pareto optimality, approximate, online, parallel, critical response time, high risk" [7].

- *DSS Architectonics*. The software process (not package) is "vaguely specified, validated against end-user expectations" and has "two new fundamental design-space dimensions: Time and Uncertainty" [7].

- *IT infrastructure and paradigms*. "The IT infrastructure is sufficiently advanced (in both facts and trends: nanoelectronics, broadband communication, semantic web, multimodal interfaces, etc.) to allow anthropocentrism for most IT applications" [8]. The leading paradigm is "computing as interaction" [1] [19]. Second echelon paradigms (prevalent in modern artificial intelligence) are assessed through an "affordability filter" in [6]. As regards software engineering, the paradigm that becomes dominant is agent-orientation.

Diagnosis. The main weaknesses of current IT systems are investigated in [7], where the focus is on conventional modelling. Since DSS weaknesses are very similar, they are stated here adapted and abridged from [7]: they stem from inappropriate conceptualising, based on rigid, algorithmic (i.e., deterministic, almost sequential, "computational", and atemporal processing), meant for decision making as "step by step solving of arising sub-problems", not for decision making as "continuous process of dealing with unexpected, potentially risky, fast changing situations requesting immediate - albeit not optimal - response". Sectorial aspects are:

- *Poorly reflected* (or absent) temporal dimension. Limited parallelism (if any), ineffective multithreading, poor reactivity (scarce interrupt handling impairing proper reaction to environment stimuli); no exception propagation; no dynamic priorities; no proper thread synchronization). Since the agent is a process - now acknowledged as such by a formal standard [13] - its temporality cannot be disregarded anymore.

- *Poor concurrent programming*. Often such situations are handled by resource wasting "active wait loops" or counterproductive "mutexes" instead of a simple Wait (for an event) with Timeout.

- *Misunderstood uncertainty*. Even if the fact that accurate numeric data are hard to get is accepted, the emphasis is on approximated, predicted, evaluated by rule of thumb, or even on intrinsically fuzzy data, rather than on missing ones (lacking sensor information, delayed previous decisions, server crash etc.).

- *Distorted prediction*. Bayesian inferences are considered unsuitable to decision-making because "Even if decision-makers could get all [Ĕ] answers in due time, would they believe them strongly enough to make critical decisions only on their basis? Humans are not "probabilistic beings" and are very prone to any sort of "gambler's fallacy"." [9].

In short, the real-world decision-making challenge is: the situations are such that you have no time to solve (accurately, complex) problems.

3 Delimiting the Problem

The multifaceted issue of handling uncertainty in DSS must be first restricted to arrive at manageable complexity and afterwards defined as software engineering task.

Restricting the scope. The target was narrowed to deal with undecidability due to future contingents because of five reasons:

- *Unaffordable complexity*. Uncertainty as epistemic concept, together with its species and degrees, was investigated in [9] starting from the 28 definitions found on the Web. Beside the overwhelming diversity of those definitions, ranging from "doubt" to "statistically defined discrepancy", the very meaning of uncertainty "depends on the professional background and on the task to carry out (better said, mostly on the time available to complete it)" [9]. Thus, "uncertain" means practically (mainly, subconsciously) for mathematicians, unknowable, for software developers, undependable, and for end users (decision-makers), undecidable. Recent related work attests the link between uncertainty and complexity: a terminology and typology of uncertainty "together with the role of uncertainty at different stages in the modelling processes. Brief reviews have been made of 14 different (partly complementary) methods commonly used in uncertainty assessment and characterisation" is presented in [17]. A rare case when undecidability is discussed outside mathematics is in a sociological context where the authors "suggest that as well as being able to consider organizational decision-making as an instance of (albeit bounded) rationality or calculability, it can

also be regarded as a process of choice amongst heterogeneous possibilities" [12]. "In this context could be found a common denominator for a general definition of uncertainty - at least, acceptable to the three categories mentioned above? Uncertainty, in its widest sense, comprises any unsure link in the chain of steps necessary to fulfil a task" [9]. Much too complex to cope with.

- *Avoidable complexity.* On the other hand, from the 28 definitions "only a few are interesting, since they are anthropocentric, mirroring the common user (mainly decision-maker) stance: a) "doubt" [...]; b) "the fundamental inability to make a deterministic prognosis" [...]; c) "lack of knowledge of future events"." [9].

- *Intrinsic importance*. "Real-world problems show that the most important and ill-treated kind of uncertainty is that due to future contingents: decisions are difficult to make because a relevant event not happened yet, not because a result is imprecise. Moreover, its pragmatic corollary highlights a key aspect in decision-making: since any statement about a future event is undecidable, how to proceed in this case? Should it be predicted, circumvented, waited?" [9].

- *Time Pressure*. The geometrically increasing computing power due to Moore's law (mentioned in Section 1) promotes factors tending to reduce radically the role of algorithms and (bivalent) logic in IT [7]. Two reasons are already manifest: "Since deterministic applications are vanishing (because of OHDUE), the conventional algorithm is not anymore program backbone. Even when still useful, the conventional algorithm is not anymore the main programming instrument (being hidden in procedures easily reached in a host of libraries or being generated by 4GL)" [8].

- *Risky Side Effects*. An algorithm is almost unable to feel time: no sense of future events, no such step as: "Warning: I don't know yet". Worse: often the algorithm cheats, confusing undecidability with negation.

Defining the task. "Since here the issue is to design a mechanism not a particular application, the cardinal concern, from a clear-cut software engineering perspective, is about reducing complexity, both structural (to make the mechanism useful to legacy systems too) and cognitive (to motivate system designers as well as to increase user acceptance)" [9]. Software engineering constraints imply: simple tools, with immediate applicability in current designs; no sophisticated concepts or instruments (such as agents, temporal logics, explicit uncertain information processing, computability theory and computational complexity theory, Bayesian methods, certainty factors, etc.); bottom-up design and testing; as much as possible conventional development (prevalent algorithmic reasoning, usual API functions, straightforward implementation, downward compatibility, etc.).

While most restrictions are comprehensible - albeit very tough -, prohibiting both temporal logics and explicit uncertain information processing seems counterproductive, since one of the premises claimed Time and Uncertainty as fundamental design-space dimensions. Unfortunately, even most responsive and appropriate approaches ([14] included) are less applicable because they are sectorial (e.g., treating time without uncertainty or vice versa). Other interesting temporal logics are too sophisticated: for instance, the "linear time" logic (used for specifying reactive systems) is based on the two modalities "Since" and "Until"; in [16], extensions of this logic are investigated from the perspective of undecidability for "X will happen within t unit of time" showing that the extension is undecidable, whenever t is irrational.

Although aspects of a primitive temporal dimension could be implemented using common API functions - e.g., Wait (for an event) with Timeout - as well as synchronization methods used in multithreading, no similar mechanism is available for uncertainty. Here lies the innovative core of the undertaking with all its potential, openings and risks: the only mechanism on hand in common operating systems, able to shift initiative from algorithm to environment, is exception handling. Thus, the algorithm gives up a morsel of its proactiveness - deterministic par excellence - to gain a touch of reactivity; this "small

amount of non-determinism" is mandatory to be able to mirror (in real time) its epistemic facet: uncertainty. (In DOMINO, when a conditional expression in IF is undecidable, an exception is raised and control is handed over to its handler.)

The idea to assign a new semantic value - i.e., enabling a software entity (labelled or not as agent) to react prompt to asynchronous environment stimuli - to a mechanism meant to increase robustness was advanced (and defended in detail) in [3] after testing it in expressing emotions of pathematic agents. Later, using exceptions to achieve agent reactivity was suggested in the context of sub-symbolic inferences [8] and of emergence as leverage [9]: "Even primeval animals move "algorithmically" ("if gap then avoid, else go on") only a few steps, in very hostile environments. Moreover, reaction to stimuli cannot mean perpetual looking for the stimulus. The cardinal hindrance stems [...] from the mechanisms employed: neither nature, nor technology can afford in the long run mechanisms involving large amount of testing because they are too time-consuming tools". (Currently exceptions are tested to help self-referencing agents to show some primitive form of self-awareness [5]. However, in recent related work exceptions are still regarded solely as a "mechanism for structuring error recovery in software systems so that errors can be more easily detected, signaled, and handled" [11]. One of the "fundamental motivations for employing exception handling in [...] robust applications" is to "improve the maintainability of the normal code and promote reuse of error handling code" [11]. In [10], the exception handling mechanism of a state-of-the-art industrial embedded software system is analysed and the fault-proneness of the return-code idiom - for dealing with exceptions - is evaluated.)

As a result, since a conditional expression in an ordinary IF has now a third exit variant (the exception), the very IF acquires a trivalent logic semantics. Then, why stay at the stage of "trivalent logic semantics" and not go further towards "trivalent logic"? Because such logics are far to meet the challenge, despite the huge effort to conceptualise, develop and implement them. A three-valued logic is a "logical structure which does not assume the law of the excluded middle. Three truth values are possible: true, false, or undecided. There are 3072 such logics" [18]. To avoid the 3073rd one[1] , "for the sake of simplicity, the trivalent semantics should be grounded on a usual bivalent software infrastructure" [9]. Indeed, the proposed solution for DOMINO considers the revisited concept of undecidability [7] [9] filtered through the double sieve of relevance to usual decision-making (Section 4) and design simplicity as vital software engineering request (Section 5).

4 In Search of the Third Value Semantics

Since the key aspect in decision-making is to handle "don't know"-like uncertainty [7] (e.g., when a relevant component of an IF condition is undecidable because an expected event not happened yet), it is appropriate to a third value meaning something like "Caveat: I don't know (yet)", "unknowable" or "unknown". In many-valued logics it "is general usage [...] to assume that there are two particular truth degrees, usually denoted by "0" and "1", respectively, which act like the traditional truth values "falsum" and "verum"." [15]. "Obviously, any decision-making needs those pillars of bivalence" [9]. Thus, the third value should be added to the two Boolean constants of IF. In short, the three outputs are: *Yes, No, Undecidable*, where it must still be decided what should "*Undecidable*" really mean.

The investigation starts with the first and more relevant three-valued logics and their respective motivations and meanings of the third truth value:

- *Łukasiewicz.* The first intention was to use the third value for "possible" (to model modalities) to deal with future contingents (first of all, the "paradox of the sea battle" posed by Diodorus Cronus). Ignoring the philosophical motivation and context (the ontological status of the future, is it determined or not, does free will actually exist, etc.), the third value - "**i**" for "indeterminate", interpreted

[1] At least (supposing that in this century no other trivalent logic has been concocted)

as "unknowable" or "problematical" - is semantically very close to the real-world decision-making problems. (On the contrary, the "1/2" notation - although elegant from a (meta)mathematical perspective - is totally unacceptable in software because, for both theoretical and practical reasons, interpreting it as intermediate value of "half true and half false" is at least confusing and useless; moreover, it could lead to computational disaster.)

- *Kleene.* The third value in this logic is "**u**", interpreted as "undefined" or "undetermined" or "unknown" - with two connotations: "permanently unknown" or "temporary lack of knowledge". The second meaning is helpful for postponing decisions and is actually used in data base applications. For instance, "SQL implements ternary logic as a means of handling NULL field content. SQL uses NULL to represent missing data in a database. If a field contains no defined value, SQL assumes this means that an actual value exists, but that value is not currently recorded in the database. [...] Comparing anything to NULL - even another NULL - results in an UNKNOWN truth state. For example, the SQL expression "City = 'Paris'" resolves to FALSE for a record with "Chicago" in the City field, but it resolves to UNKNOWN for a record with a NULL City field. In other words, to SQL, an undefined field represents potentially any possible value: a missing city might or might not represent Paris." (en.wikipedia.org/wiki/ Ternary_logic). Moreover, the Kleene logic, as a natural sublogic of Belnap's four-valued logic [2], is an important framework for many paraconsistent logics - major feature for application in computer science.

- *Bochvar.* Inspired by the examination of semantic paradoxes, the third truth value "**m**" means "meaningless" or "paradoxical". This logic is useful when syntactic meaningfulness (rather than semantic one) is looked for (crucial for program verification but dispensable for decision-making).

Hence the semantics of the third value in a "three-output IF" should be based on a blend of a Łukasiewicz "**i**" interpreted as "unknowable" or "problematical"[2] and a Kleene "**u**" interpreted as "temporary lack of knowledge". Thus, the semantics of "Undecidable" is refined to "Undecidable in the time span given". In fact, it gives a chance to the "yet" in "I don't know (yet)", postponing the verdict of "Undecidable" as much as possible (depending heavily on both the problem to solve and the decision-making strategies applied). Of course, to be effective, such a procrastination strategy must be put to work only in a distributed environment (reflected in software through multithreading); otherwise, the user would be frustrated by the frequent wait periods.

5 In Search of Bivalent Syntax Clothes

DOMINO respects the rigours of structural programming and the syntax of bivalent logic, is programmed in plain C++ (to be applicable to legacy systems too), is based on a few functions of the Windows32 API, was tested only within the toy problems (regarding real-time decisions in industrial control or in medical informatics) the examples are taken from, and is currently tested in a real-world problem (a simplified version of implementing overbooking policy). Its flowchart is in Figure 1 (the only symbol that is not among the common flowchart notation is the one for raising exceptions).

The (Windows 2000) API functions involved are:

```
a)
BOOL SetEvent(hEvent)
HANDLE hEvent;     /* Event-object handle */
   Example: SetEvent(g_hEvent_pHLimit);
```

[2] According to the ancient Stoic perspective, updated by Łukasiewicz

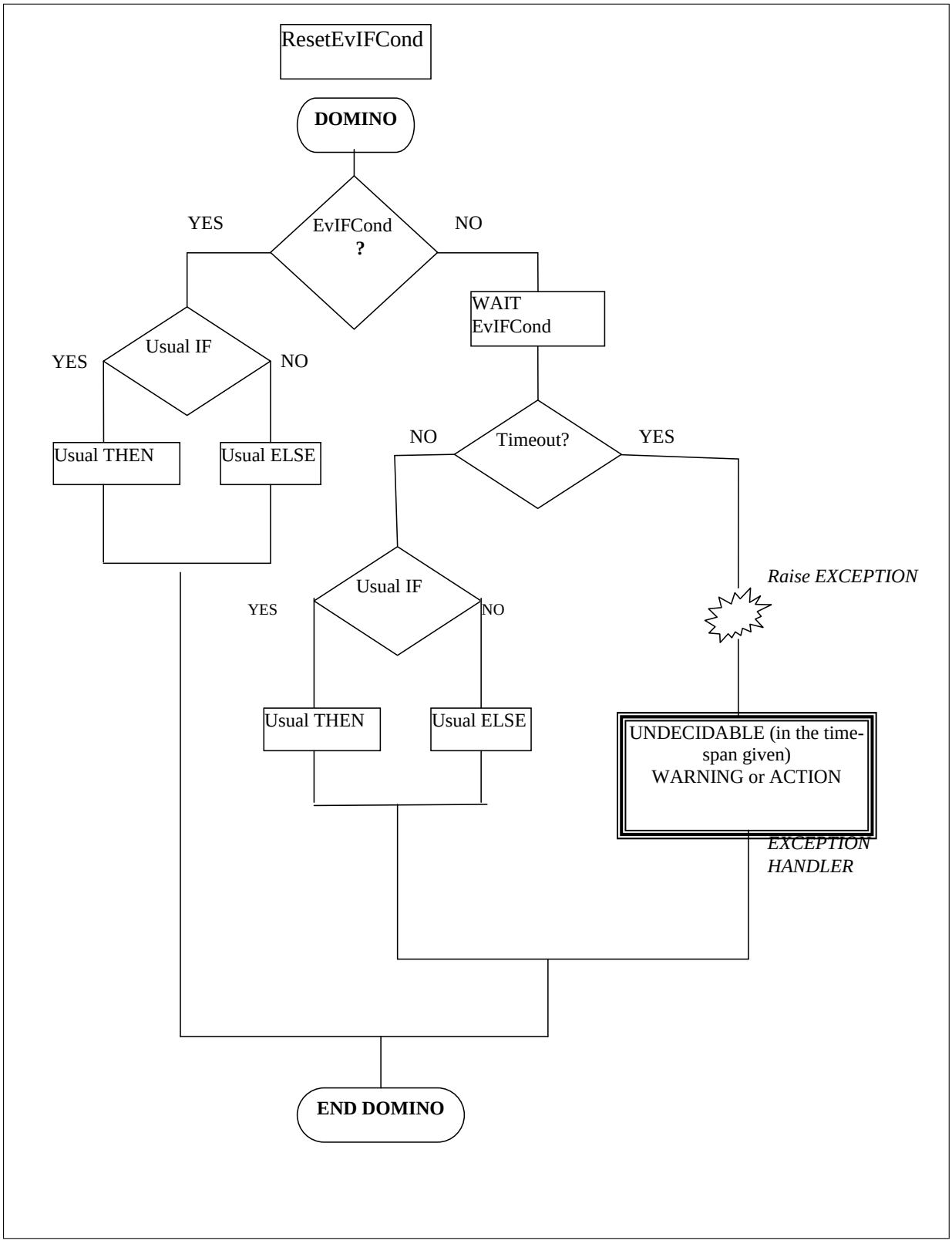


Figure 1: DOMINO: Architecture expressed in bivalent logic

```
b)
BOOL ResetEvent(hEvent)      /* likewise */

c)
DWORD WaitForSingleObject(hObject, dwTimeout);
HANDLE hObject;  /* handle of the Event waited for*/
DWORD dwTimeout; /* maximal Wait duration (ms)
(INFINITE: unlimited wait) (0: testing the Event state) */
    Example from an alarm-bell program:
dwResult = WaitForSingleObject(g_hEventOverheating, INFINITE);

d)
DWORD WaitForMultipleObjects(cObjects, lphObjects,
                                    fWaitAll, dwTimeout);
DWORD cObjects;  /* number of objects (maximum 64) */
CONST HANDLE *lphObjects;      /* handle table address */
BOOL fWaitAll;   /* flag for conjunction /disjunction
  (TRUE: all events are waited)
  (FALSE: only the first event is waited)*/
DWORD dwTimeout;         /* likewise*/
    Example from a domotics program
    (for 5 boilers; uses multiple events):
dwResult = WaitForMultipleObjects(5, hBoilerPressure, TRUE, 4000);
```

Remarks.

1. The exception handler can: a) give control to the human decision maker (after a warning); or b) act (for instance, propagating dynamically the exception from the callee to the caller).

2. Time is not just explicitly present in "Wait" but, more important, hidden in thread synchronization (SetEvent is employed in the thread providing information needed to make a decision).

3. Uncertainty is dealt with through the intrinsically uncertain interrupts generated by the environment stimuli, expressed at the program level through the exception handler.

4. If dwTimeout = 0, the mechanism is a conventional "IF" (however, a more robust one since evaluation is preceded by a real-time validation test).

6 Expectations

The three time horizons are roughly delimited by: a) validating DOMINO in the overbooking application; b) designing and validating other (similar) mechanisms of the AGORITHM toolbox [9]; c) API functions for DOMINO-like primitives.

Short range.

A) DOMINO addresses the main problem of current "IF" in applications running in OHDUE: "IF" is inadequate not because it cannot say "probably 80B) It solves this problem in a simple way (for both designer and user).

C) It is easy to implement due to its straightforward structure involving only common API functions: despite its enriched semantics the emulated "IF" does not need a modified compiler.

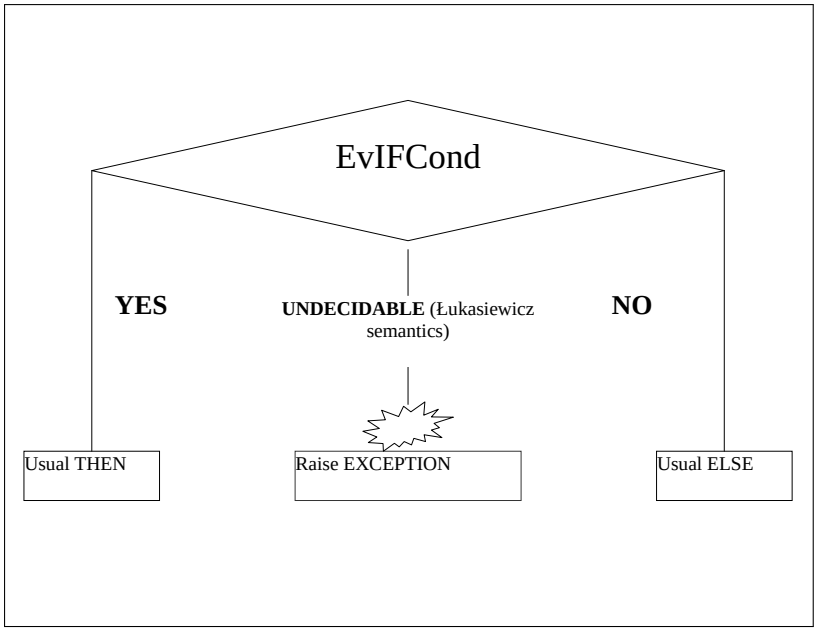


Figure 2: DOMINO: Semantics interpreted in trivalent logic

Middle range.
A) Corollary 1: being downward compatible, it should be useful for legacy systems.
B) Corollary 2: emulating a language primitive, it should be useful not only for DSS, but for any application in OHDUE.
C) Corollary 3: dealing with both time and uncertainty, it should be useful for any agent-oriented application: other AGORITHM mechanisms could be outlined similarly.
D) Hopefully it will start a "virtuous circle" able to adapt the DSS to the "Information Age" requirements changing both the decision-makers expectations and the programming paradigm.

Long range.
A) A new interpretation of DOMINO: a three-valued IF within the frame of bivalent logic
B) The badly needed shift in software engineering towards exception handling: the stimulus causes an interrupt that is treated as exception. (Real-world applications cannot afford large amount of "if temperature > n 0C then alarm" testing
C) Motivating applied research in AI logics, avoiding the gap between new logics (dealing at highly abstract mathematical levels with either time or non-determinism but rarely with both) and decision-support applications (either remaining at the level of rigid algorithmic approach or using approaches that ignore the fundamentals of human decision making).
D) A new, "procrastination logic": less algorithmic, less probabilistic, less difficult to understand and to apply in OHDUE, where postponing a decision means keeping open alternatives (to react better to rapid environment changes).

Bibliography

[1] AgentLink Roadmap: Overview and Consultation Report, AgentLink III. Agent based computing, University of Southampton, 2004.

[2] Anderson, A.R, N.D. Belnap, J.M. Dunn. *Entailment: The Logic of Relevance and Necessity*, Volume 2. Princeton University Press, Princeton, 1992.

[3] Barbat, B.E. *Agent-Oriented Intelligent Systems.* Romanian Academy Publishing House, Bucharest, 2002 (in Romanian, "Grigore Moisil" Prize of the Romanian Academy).

[4] Barbat, B.E. *The Impact of Broad-Band Communication upon HMI Language(s). (Chapter 7.) Communicating in the world of humans and ICTs. (Chapter 8.) in COST Action 269. e-Citizens in the Arena of Social and Political Communication* (L. Fortunati, Ed.), pp. 113-142, EUR21803, Office for Official Publications of the European Communities, Luxembourg, 2005.

[5] Barbat, B.E., A. Moiceanu, I. Pah. *Gödelian Self-Reference in Agent-Oriented Software.* Proc. of the 11th WSEAS International Conference on COMPUTERS (ICCOMP '07) (N.E. Mastorakis et al, Eds.), 92-97, Agios Nikolaos, Crete, 2007.

[6] Barbat, B.E, A. Moiceanu, S. Plesca, S.C. Negulescu. *Affordability and Paradigms in Agent-Based Systems.* Computer Science Journal of Moldova, 2007. (In print.)

[7] Barbat, B.E., R.S. Muntean, R. Fabian. *Approximation versus Undecidability in Economic Modelling.* Proc. of the International Workshop New approaches, Algorithms and Advanced Computational Techniques in Approximation Theory and its Applications (D. Simian, Ed.), 2007. (In print.)

[8] Barbat, B.E., S.C. Negulescu. *From Algorithms to (Sub-)Symbolic Inferences in Multi-Agent Systems.* International Journal of Computers, Communications & Control, 1, 3, 5-12, 2006. (Paper selected from the Proc. of ICCCC 2006.)

[9] Barbat, B.E., S.C. Negulescu, S. Plesca. Emergence as Leverage and Non-Algorithmic Approaches in Agent-Oriented Software. Studies in Informatics and Control Journal, 16, 4, 2007. (In print.)

[10] Bruntink, M., A. van Deursen, T. Tourwé. *Discovering faults in idiom-based exception handling.* Proc. of the 28th international conference on Software engineering, 242 - 251, ACM Press, New York, 2006.

[11] Castor Filho, F., A. Garcia, C.M.F. Rubira. *Error Handling as an Aspect.* Proceedings of the 2nd workshop on Best practices in applying aspect-oriented software development, ACM Press, New York, 2007.

[12] Clegg, S., Kornberger, M., Rhodes, C. *Organizational ethics, decision making, undecidability.* The Sociological Review, 55, 2, 393-409(17), Blackwell Publishing, 2007.

[13] FIPA TC Agent Management. *FIPA Agent Management Specification. Standard SC00023K (2004/18/03).* http://www.fipa.org/specs/fipa00023/SC00023K.pdf, 2004.

[14] Fisher, M. *Temporal Development Methods for Agent-Based Systems. Autonomous Agents and Multi-Agent Systems,* 10, 41-66, Springer Science + Business Media Inc., 2005.

[15] Gottwald, S. *Many-valued Logic. In Stanford Encyclopedia of Philosophy* (E.N. Zalta, Ed.). http://plato. stanford.edu/entries/logic-manyvalued/, 2004.

[16] Rabinovich, A. *Temporal logics with incommensurable distances are undecidable.* Information and Computation, 205, 5, 707-715, Elsevier, 2007.

[17] Refsgaard, J.C. et al. *Uncertainty in the environmental modelling process-A framework and guidance.* Environmental Modelling & Software, 1543-1556, Elsevier, 2007.

[18] Weisstein, E.W. Three-Valued Logic. *From MathWorld–A Wolfram Web Resource.* http://mathworld. Wolfram.com/Three-ValuedLogic.html. CRC Press LLC, Wolfram Research, 1999.

[19] Zambonelli, F., A. Omicini. *Challenges and Research Directions in Agent-Oriented Software Engineering.* Autonomous Agents and Multi-Agent Systems, 9, 253-283, Kluwer Academic Publishers, 2004.

Boldur E. Bărbat
"Lucian Blaga" University of Sibiu
Department of Computer Science
5-7 Ion Raţiu St., 550012, Sibiu, ROMANIA
Email: bbarbat@gmail.com

Received: March, 24, 2007

Boldur Barbat M.Sc. in Electronic Engineering, postgraduate specialising in Programming, Ph.D. in Digital Computers ("Politehnica" University Bucharest). He is with "Lucian Blaga" University Sibiu, Faculty of Sciences (full professor) and "Politehnica" University Timisoara, Faculty of Automation and Computers (advisor for doctoral studies in Computer Science). Over 20 books (Romanian Academy IT Prize, 2002). About 50 papers/articles in English in the last five years. Current research interests: emergence in agent-based systems (self-awareness, stigmergy), human-agent interaction (transdisciplinary), agent-oriented software engineering (non-algorithmic real-time software, uncertainty).

International Journal of Computers, Communications & Control
Vol. II (2007), No. 4, pp. 314-327

An Implementation of Reconfigurable Network Control based upon Automata Proposal for Three Conveyor Belt Case Study

Benítez-Pérez H., Cárdenas-Flores F., García-Nocetti F.

Abstract: Online reconfiguration performed by a computer network system needs to be addressed from several perspectives due to complexity onto the system. This paper proposes different modeling approximations to obtain a holistic view of reconfiguration onto complex systems. First model is dynamic system modeling, second is an automaton in order to bound possible scenarios and third model is a Real Time scheduling algorithm to match possible configurations and related control laws.

1 Introduction

One of the main issues in fault tolerance is to keep availability even in hazard situations. A way to guarantee this is by reconfiguration where several consequences are expected. In that respect, either fault coverage, masking or tolerance are strategies to use during reconfiguration. Reconfiguration is a need in order to keep safety during fault scenarios. The results of this action modify several structures within complex systems like communication, dynamic behaviour and predictable response. In order to cover every aspect of these requirements, modeling becomes a crucial issue to get enough information by performing reconfiguration. In here, three different modeling techniques are followed, dynamic system modeling, automaton modeling and scheduling representation. These three strategies, by there own, lacks of a holistic view of the effects of a process such as reconfiguration. For instance, automaton strategy allows a structural view of reconfiguration without determining the effects of this action into dynamic behaviour of the system. Alternatively, dynamic system modeling provides a formal view of the effects of time delays and the loose of certain dynamic elements, however, reconfiguration is not a predetermined action only the effects of it. The novelty of this approximation is to integrate various modeling strategies to accomplish reconfiguration and its effects.

Firstly, dynamic system is modeled in order to determine where time delays play a key role. Secondly, automaton modeling is pursued in order to bound scenarios during fault and fault-free situations. From this automaton representation and taking into account real-time requirements, real-time scheduling through a special representation is implemented. The combination of these three allows system modeling even in complex situations like reconfiguration.

The objective of this paper is to present a strategy for control reconfiguration based upon time delay knowledge using a scheduling algorithm. Fault effects are local within a distributed system environment. The use of a case study is pursued to accomplish this objective. The novelty of this work is the amalgamation of scheduling and control techniques to get this strategy.

In particular, for the case of dynamic system modeling, several strategies for managing time delay within control laws have been studied for different research groups. For instance Nilsson [8] proposes the use of a time delay scheme integrated to a reconfigurable control strategy based upon a stochastic methodology. On the other hand, Wu [10] proposes a reconfiguration strategy based upon a performance measure from a parameter estimation fault diagnosis procedure. Another strategy has been proposed by Jiang et al. [6] where time delays are used as uncertainties, which modify pole placement of a robust control law. Izadi et al. [5] present an interesting view of fault tolerant control approach related to time delay coupling. Reconfigurable control has been studied from the point of view of structural modification since fault appearance as presented by Blanke et al. [2] where a logical relation between dynamic variables and faults are established. Alternatively reconfigurable control may performs a combined modification of system structure as studied by Benítez-Pérez et al. [1] and Thompson [9]. Another technique

like gain scheduling (Khalil, [7]) may give an interesting approximation to several time delay scenarios, however complexity related to system modeling during fault conditions is out the scope of this paper.

Some considerations need to be stated in order to define this approach. Firstly, faults are strictly local in peripheral elements and these are tackled by just eliminating the faulty element. In fact, faults are catastrophic and local. Time delays are bounded and restrictive to scheduling algorithms. Global stability can be reached by using classical control strategy for online time delays.

2 Case Study and Control Reconfiguration Approach

Due to the complexity of this approach a particular implementation is pursued in order to present the potential of complementary modeling for reconfiguration. The Case study is based on three conveyor belts (Gudmundsson, [4]) integrated as follows, it comprises 3 conveyors belts, 4 actuators, and 12 sensors. It has 16 computing elements that considers the controller and the bus controller. Fig. 1 shows a diagram of this implementation. The procedure of the example is next; conveyor belt 1 detects a box, it modifies its speed up to a certain level to transport it in a faster way the box. MC is stated for Micro-Controller

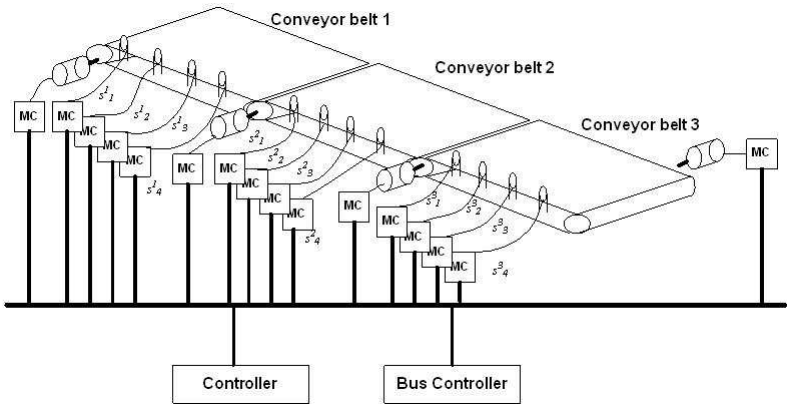


Figure 1: **Conveyor belt example**

When this box arrives to conveyor belt 2, its speed is modified up to another level to transport this box in a faster manner. Similar behavior is presented at conveyor belt 3. The sensor vector is used to detect the current position of box in any of these conveyor belts. Furthermore, actuator 4 has the task of pushing the current box when it arrives at this position.

For the case of multiple boxes in this example, the aim is that the boxes never crashed between each other. From this explanation, Table 1 shows the modification of speeds.

This case study is peculiar in that each conveyor belt has two different speeds as shown in Table 1.

These speeds are dependent on the sensor situation. This sensor situation is depicted as low and high, which is a semaphore for determining the presence of an object.

The second peculiarity is related to the difference between HS as follows:

$$HS^1 < HS^3 < HS^2, \tag{1}$$

where the middle conveyor belt is the fastest, then third conveyor belt, and so on. As the reader may realize, there are four motors, three for the conveyor belts and the fourth is to pull any object presented at its region.

Based on this case study, the response of the three actuators is shown in Fig. 2. Different speed-ups are shown assuming that a box is presented during a certain time. For instance, the first conveyor belt

Table 1: Speed selection

	Conveyor belt 1	Conveyor belt 2	Conveyor belt 3
Sensors $S_*^1 = \texttt{Low}$	LowSpeed	Low Speed	Low Speed
Sensors $S_*^1 = \texttt{High}$	High Speed HS^1	Low Speed	Low Speed
Sensors $S_*^2 = \texttt{Low}$	Low Speed	Low Speed	Low Speed
Sensors $S_*^2 = \texttt{High}$	Low Speed	High Speed HS^2	Low Speed
Sensors $S_*^3 = \texttt{Low}$	Low Speed	Low Speed	Low Speed
Sensors $S_*^3 = \texttt{High}$	Low Speed	Low Speed	High Speed HS^3

presents a faster speed-up during the first 3000 seconds, in comparison with the low speed-up during 3000 to 6000 seconds. This speed-up is shown as a change of slope of the current graphic. Similar behavior is presented for both conveyor belts as 2 and 3 are modified, because HS^2 is bigger than HS^3.

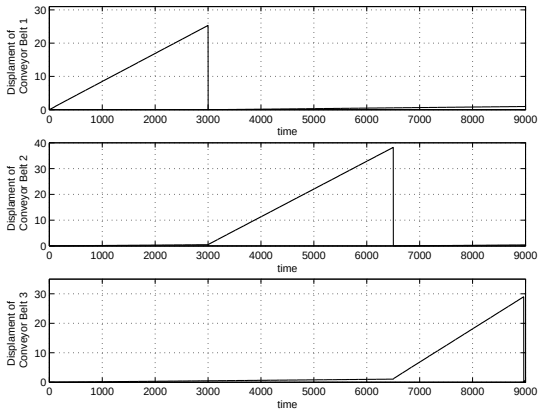


Figure 2: **Related displacement when a box is present in each conveyor belt**

2.1 First modeling approach

First modeling approach is based on control law modification taken into account time delay appearance, in particular for current case study. The schematic setup is based on Fig. 3 considering system response and control implementation.

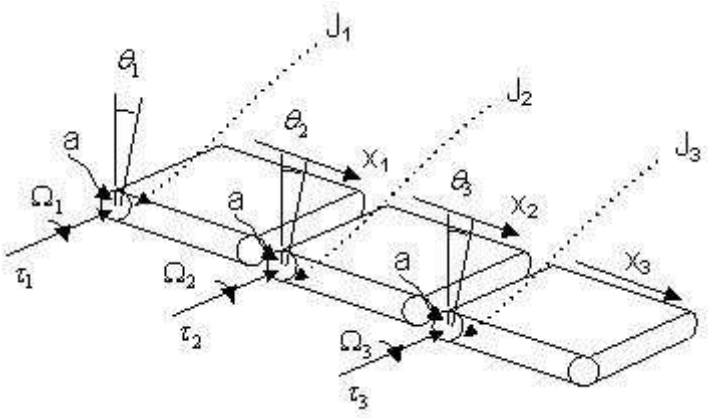


Figure 3: **Dynamic System Implementation**

In Fig. 3,

x	is the linear displacement
θ_*	is the angular displacement
ω_*	is the angular velocity
F	is the lineal force
J_*	is the lineal inercy
k	is the parameter
τ_*	is the torque
a	is the radius
v	is the lineal velocity

In this case, the plant presents two cases with or without a box per belt. As the second case is trivial, the first case is expressed per belt considering the mass of the box (referred to as m). The first conveyor belt is expressed as

$$\begin{bmatrix} \ddot{x_1} \\ \dot{\theta_1} \end{bmatrix} = \begin{bmatrix} J/m & 0 \\ & J_1 \end{bmatrix} \begin{bmatrix} \theta_1 \\ \dot{x_1} \end{bmatrix} - \begin{bmatrix} 1/m \\ 0 \end{bmatrix} \tau_1 \qquad (2)$$
$$y = \dot{x}_1.$$

The second conveyor and the third conveyor belt follow simmilar dynamics From these considerations, discrete plants are defined next by considering the presence of the box

$$x(k+1) = Ax(k) + \sum_{i=0}^{l} B_i^k u(k-i)$$
$$B_i^k = \int_{t_i^k}^{t_{i-1}^k} \exp(A(T-\tau)Bd\tau, \qquad (3)$$

where $l = 1$ because the maximum number of sensors with delays is just one. Therefore, the A matrix is expressed as:

$$A_* = \begin{bmatrix} \exp(J/m) & 0 \\ 0 & \exp(J_*) \end{bmatrix}, \qquad (4)$$

where T is the inherent sampling period, and t_0^k, t_1^k, and t_2^k are the related delays of the plant. For the case of local control laws, these are expressed next as:

$$\begin{aligned} x_c(k+1) &= A_c x_c(k) + B_c u_c(k) \\ y_c(k) &= C_c x_c(x - \tau_c - D_c u_c(k - \tau_c), \end{aligned} \tag{5}$$

giving the delays as a result of decomposition from sensor and actuators, which are expressed as τ_{sc} and τ_{ca} respectively. The augmenting representation is given next:

$$\begin{aligned} u_c(k) &= y_p(k - \tau_{sc}) \\ u_p &= y_c(k - \tau_{ca}), \end{aligned} \tag{6}$$

where states are augmented as:

$$z = \begin{bmatrix} x_p(k) \\ x_c(k) \end{bmatrix}, \tag{7}$$

and expressed as:

$$\begin{aligned} z(k+1) = &\begin{bmatrix} A_p & 0 \\ 0 & A_c \end{bmatrix} z(k) + \begin{bmatrix} 0 & 0 \\ B_c C_p & 0 \end{bmatrix} z(k - \tau_{sc}) + \begin{bmatrix} B_p D_c C_c & 0 \\ 0 & 0 \end{bmatrix} z(k - \tau_{sc} - \tau_{ca} - \tau_c) \\ &+ \begin{bmatrix} 0 & B_p C_c \\ 0 & 0 \end{bmatrix} z(k - \tau_{ca} - \tau_c). \end{aligned} \tag{8}$$

By modifying Eqn. 8 to define the stability of network control, the next configuration is proposed:

$$\begin{aligned} F^j &= \begin{bmatrix} A_p^j & 0 \\ 0 & A_c^j \end{bmatrix} \\ F_1^j &= \begin{bmatrix} 0 & 0 \\ B_c^j C_p^j & 0 \end{bmatrix} \qquad F2^j = \begin{bmatrix} B_p^j D_c^j C_p^j & 0 \\ 0 & 0 \end{bmatrix} \\ F_3^j &= \begin{bmatrix} 0 & B_p^j C_c^j \\ 0 & 0 \end{bmatrix}. \end{aligned}$$

Therefore, the state vector is modified for these time delays where the system is asymptotically stable based on $F^j + \sum_{i=1}^{3} F_i^j$. Based upon a single control loop is stable (Eqn. 8), it is possible to define stability for every loop as shown in Eqn. 10.

$$\tau < \frac{\sigma}{\delta \sum_{i=1}^{3} \left| F_i^j \left(F^j + \sum_{i=1}^{3} F_i^j \right) \right|}, \tag{9}$$

where τ is the maximum value from all possible time delays at all loops. The absolute value is used in order to guarantee positve response with respect to current time delay.

$$\begin{aligned} \sigma &= \frac{\lambda_{\min}(Q)}{2\lambda_{\max(P)}} \\ \delta &= \left[\frac{\lambda_{\max}(P)}{\lambda_{\min}(P)} \right]^{\frac{1}{2}}, \end{aligned}$$

where $\lambda_{\max}(P)$ and $\lambda_{\min}(Q)$ are the maximum and minimum eigenvalues of P and Q matrices respectively. $\lambda_{\min}$ and $\lambda_{\max}$ can not be complex values since P and Q are bounded to be real values. The proposed configuration is presented

$$\left(F^j + \sum_{i=1}^{3} F_i^j \right)^T P + P \left(F^j + \sum_{i=1}^{j} F_i^j \right) = -Q, \tag{10}$$

where P, Q are positive definite symmetric matrices and are eigenvalues of the matrix and where the chosen Equation is

$$V(x) = \frac{1}{2} x^{jT}(t) P x^{j}(t). \tag{11}$$

Based upon Lyapunov proposed Equation (Eqn. 11) and by its derivative as shown in Eqn. 12, where $\dot{x}$ is substituted by the enhanced representation of $\dot{z}$ which contains both states, from the plant as well as the controller.

$$\begin{aligned} \dot{V}(x) &= \frac{1}{2}\dot{z}^{jT}(t) P z^{j}(t) + \frac{1}{2} z^{jT}(t) P \dot{z}^{j}(t) \\ &\leq -\frac{1}{2} z^{jT}(t) Q z^{j}(t) + \\ &\quad \left| z^{jT} P \sum_{i=1}^{3} F_i \int_{t_i^j}^{0} \left[F^{j} z^{j}(t+\theta) + \sum_{i=1}^{3} F_i^{j} z^{j}(t - \tau_i^j + \theta) \right] d\theta \right|. \end{aligned} \tag{12}$$

From control law expression, the related time delays are defined as τ_{sc}, τ_{ca} and τ_c where their respective values will be incorporated later in this section thorugh spliting the time delay (Eqn. 8).

2.2 Second Modelling Approach

Having shown local control laws structures, second modeling approach is the global structure in terms of an automaton (Fig. 4). Where reconfiguration is expressed for the formal event manager. In this case, two states are possible with several events, which are managed by the sensor vector for each belt (first, second, and third belts) and expressed as $S^1_{1\leq i\leq N}$,$S^2_{1\leq i\leq N}$ and $S^3_{1\leq i\leq N}$, respectively considering fault free scenario. It is important to mention that $S^1_{1\leq i\leq N}$,$S^2_{1\leq i\leq N}$ and $S^3_{1\leq i\leq N}$ are indepdent conditions between conveyors, then, $S^1_{1\leq i\leq N} = 0$ means there is no box on conveyor belt 1. At the same time $S^2_{1\leq i\leq N} \neq 0$ can be stated meaning there is a box in second conveyor belt. Other condition can be presented as $S^1_{1\leq i\leq N} \neq 0$, $S^2_{1\leq i\leq N} \neq 0$ and $S^3_{1\leq i\leq N} \neq 0$ and where three boxes are presented on the system, each box per conveyor belt. The same sensor conditions are presented in Fig. 5.

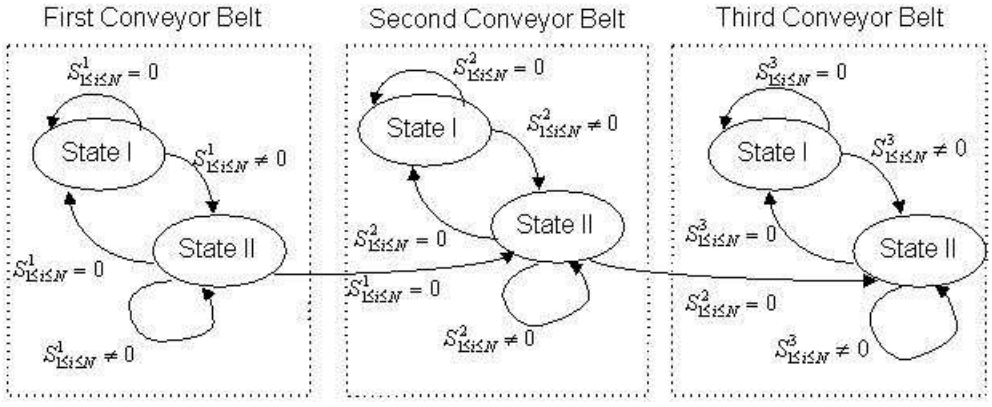


Figure 4: **Fault-free scenario in terms of global structure**

The switching effect is neglected in this fault-free scenario. In this scenario two cases are defined, when a box is presented (case II) or the other case (case I). For the second case the chosen control is to maintain the conveyor belt in zero speedup. For first case, the chosen controller is related to certain speed up depending on each conveyor belt.

For the case of a fault scenario, a new state appears for global control (Fig. 5) related to the action pursued when a fault is presented. The necessary event for reaching such a state is $S^1_{1\leq i\leq N} \neq 0$, and the fault's last event is composed of local information given by each local sensor with a relation to the health condition measures.

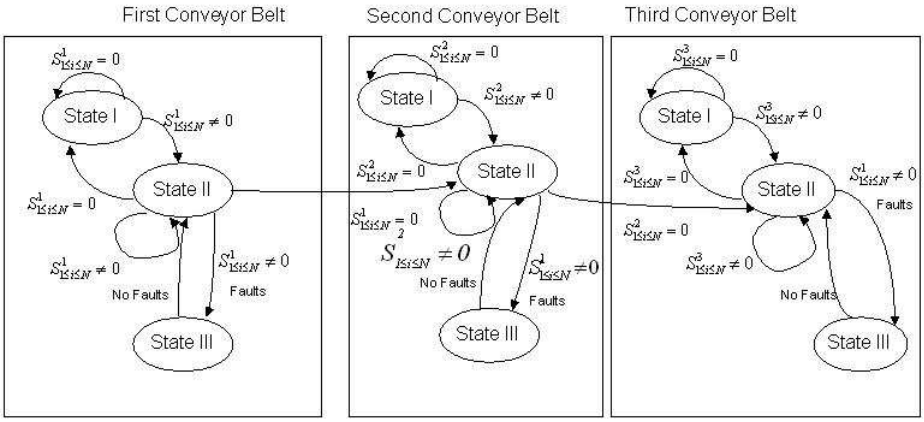


Figure 5: **Local fault scenario for the global structure**

Considering individual modeling, there is one type of local fault to be considered, which is that one of the sensors is faulty with no consideration of the type of fault. It is assumed that the fault is detectable and measurable (This is a condition in this paper).

2.3 Third Modeling Approach

Third modeling approach is related to the use of scheduling algorithm in order to determine possible time delays between processes. It is important to remember that the fault tolerance strategy is based on the use of consecutive sensors to mask the fault using extra communication to perform lateral agreement. Therefore, this approach provides two different time graphs, one for each scenario (fault and fault free), as shown in Figs. 6 and 7, respectively. The cases from Figs. 4 and 5 are related to bounded time delays from decision maker (Benítez-Pérez et al., [1] and have an effect into control and plan modeling as shown before. At the end of this section (Eqn. 15) it is shown how time delays are modified based on time diagram representation (Figs. 6 and 7). The reader should realize that time delays are bounded by the use of scheduling algorithm through ART2 network.

Both scenarios are local with respect to one belt. It is considered that the other two belts do not present faulty conditions. As these two scenarios are bounded (fault and fault-free), the respective total consumption times (tt and tt_f respectively) are shown in Eqns. 13 and 14 (Figs. 6 and 7, respectively), where variable information is presented.

$$tt = t_s * 4 + t^{sc}_{cm} + t_c + t^{ca}_{cm} + t_a \tag{13}$$

where:

- t_s is the consumed time by sensors
- t^{sc}_{cm} is the consumed time by communication between sensor and control
- t_c is the consumed time by control node
- t^{ca}_{cm} is the consumed time by communication between controller and actuator
- t_a is the consumed time by actuator

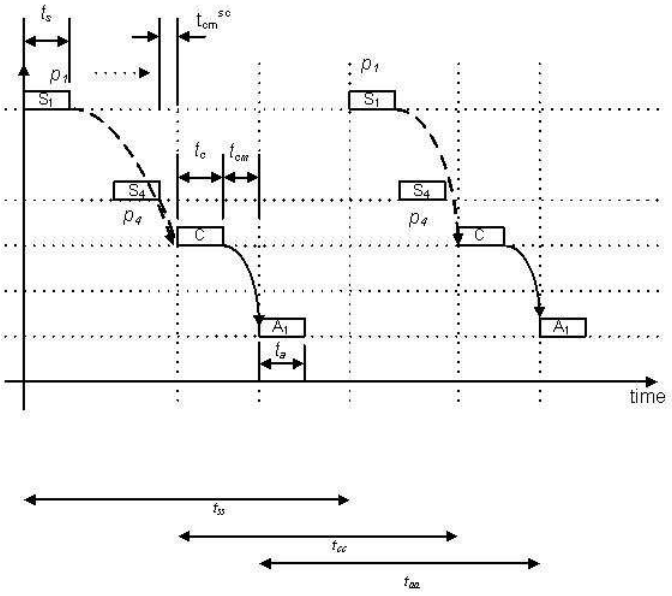


Figure 6: **Fault-free scenario**

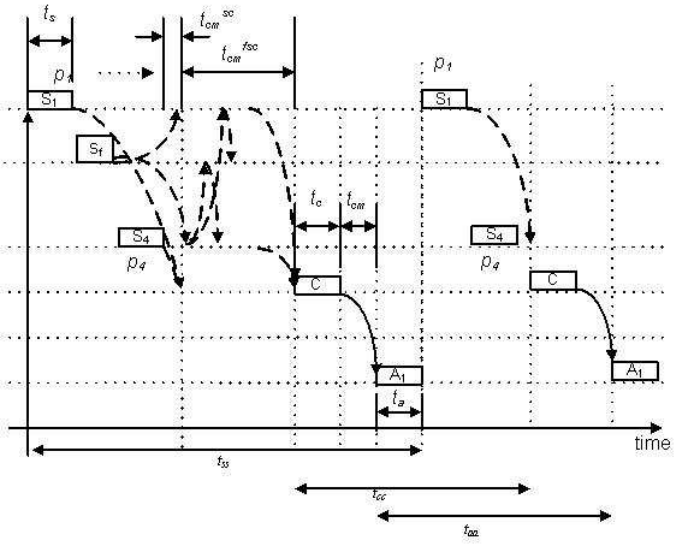


Figure 7: **Fault scenario considering fault masking**

$$tt_f = t_s * 4 + t_{cm}^{sc} + t_{cm}^{fsc} + t_c + t_{cm}^{ca} + t_a \tag{14}$$

where: t_{cm}^{fsc} is the time consumed for the fault sensor to send messages to its neighbor and produce agreement

From this time boundary based on these three modeling approaches, it is feasible to implement the control strategy as presented in Eqn. 10. As mentioned before, time delays take place in three representations τ_{sc}, τ_{ca} and τ_c , therefore, by decomposing Eqn. 14, time delays are expressed as:

$$\begin{aligned} \tau_{sc} &= t_s * 4 + t_{cm}^{sc} \\ \tau_c &= t_{cm} + t_c + t_{cm}^{ca} \\ \tau_{ca} &= t_a, \end{aligned} \tag{15}$$

A remarkable issue is related to a particular sensor fault related to any of three belts. Considering this configuration, three cases are possible:

- One local fault;
- Two local faults, one per belt;
- Three local faults, one per belt.

Based on these three possible configurations, there is a worst-case scenario related to three local faults that has an impact on the global control strategy. The other two configurations present a minor degradation for the global control strategy. Despite this performance degradation, the system keeps normal functionality due to the inherent fault tolerance strategy (fault masking between sensors) and the local controllers. Taking into account these three possible configurations, the local and global time delays are described in Table 2.

Table 2: Time delays related to local and global effects

Configuration 1	Local Time Delays	110 ms
One Local Fault	Global Time Delays	110 ms
Configuration 2	Local Time Delays	110 ms
Two Local Fault	Global Time Delays	220ms
Configuration 3	Local Time Delays	110 ms
Three Local Fault	Global Time Delays	400 ms

3 Results

From this implementation several results are presented in terms of fault presence and the related action to overcome system lack of performance. How the system responds to these control strategies is presented in the following graphics taking into account fault-free, one local fault and two local faults, respectively (Figs. 8, 9 and 10). First scenario presents a fault free situation where local controller response is shown, it is important to highlight that fourth actuator response presents a normal response.

When first fault appears (one local sensor does not response and the masking approach is followed) a local time delay takes place where its effects are shown in first local control response. Similar situation

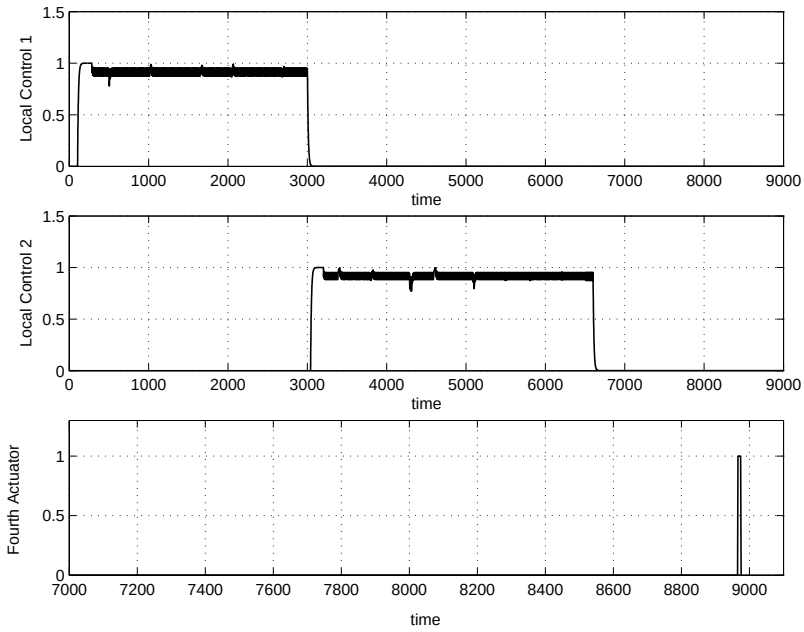


Figure 8: **Fault-free scenario**

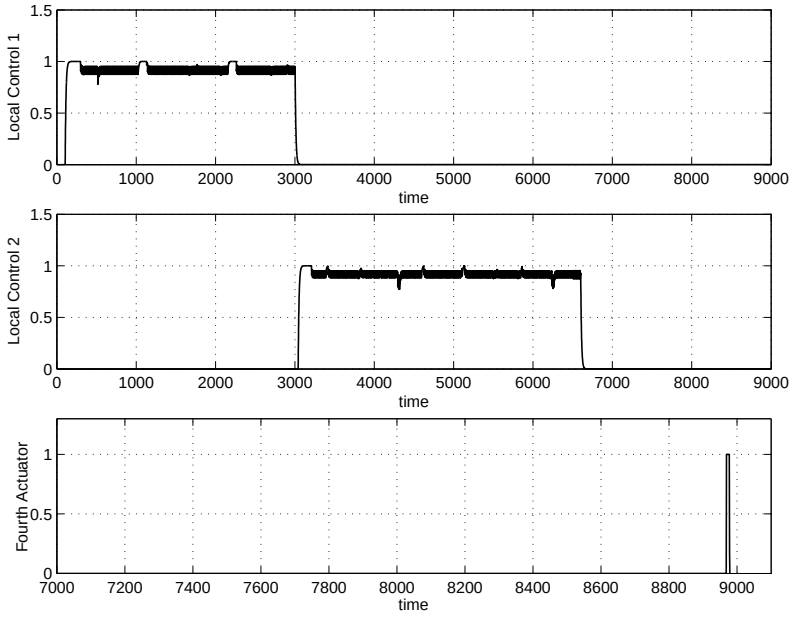


Figure 9: **First local fault appearance and related global effects**

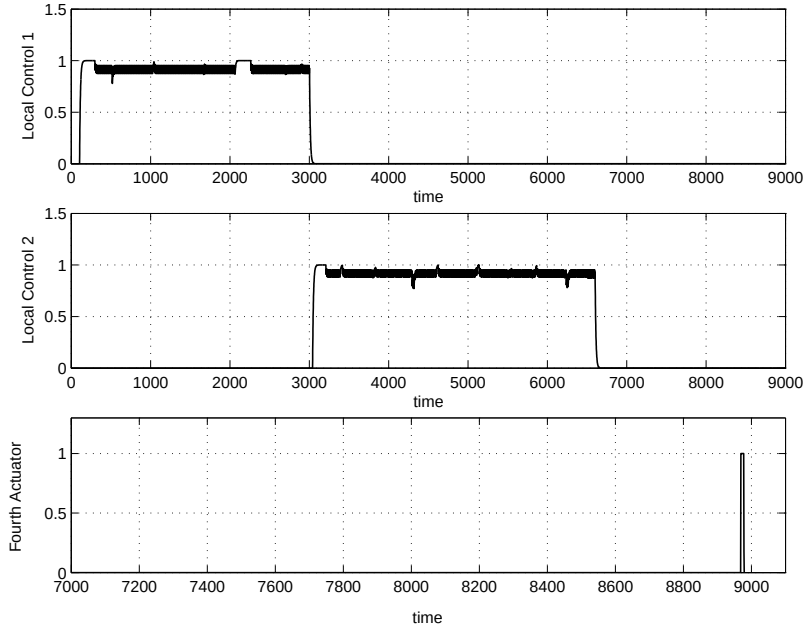


Figure 10: **Second local fault appearance and related global effects**

is presented when a second fault appears and first fault is still active (Fig. 10). In this case no relevant modification is shown due to time switching controller takes place.

This example presents three local control cases with a global automaton in which control reconfiguration is based on the scheduling algorithm, which is simple because it is dependent on the fault presence and on the related time delays. This reconfiguration approach becomes feasible due to the knowledge of fault presence and the consequence of time delays. It is obvious that fault presence is measurable; if this local fault localization approach cannot detect faults, this strategy becomes useless.

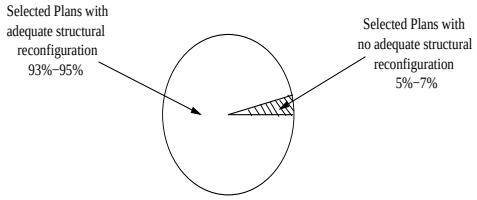


Figure 11: **Percentage of Selected Valid Plans for Structural Reconfiguration**

Moreover, local time delay management refers to the use of a quasi-dynamic scheduler to propose dynamic reconfiguration based on current system behavior rather than on predefined scenarios. The scheduler performs task reorganization based on their consumption times and fault presence.

The number of accepted plans is presented taking into those selected with no adequate response from structural reconfiguration (Fig. 11). For instance, some tasks would not have enough time to be sampled and executed. This result is presented as the percentage of the adequate use of structural reconfiguration during on-line stage. In this case, current control law is modified according to time delays status.

Having defined the percentage related to those adequate plans during structural reconfiguration, this is taking as 100 % and is evaluated in terms of control law performance. The results are presented in Fig. 12. In here, 97% of the valid plans have a valid response in terms of the mean square error response

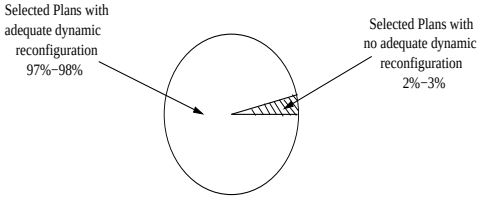


Figure 12: **Percentage of Selected Valid Plans for Control Law Reconfiguration**

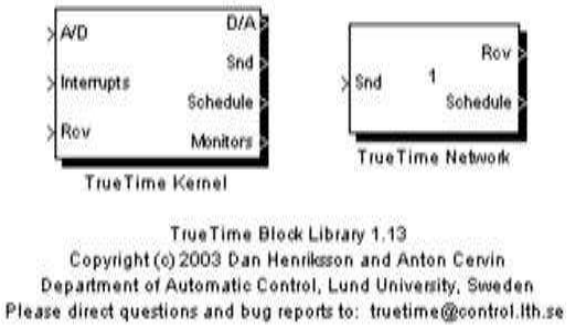


Figure 13: **Basic model of true time**

from the dynamic response of case study.

To define the communication network performance, the use of the true-time network is pursued. This strategy achieves network simulation based on message transactions that are based on the real-time toolbox from MATLAB. Extended information from this tool is available at (Cervin et al. [3]) the true time main characteristics are shown next. In the true time model, computer and network blocks are introduced in Fig. 13.

These blocks are event driven, and the scheduling algorithm is managed by the user independently of each computer block. True time simulation blocks are basically two blocks. These have been developed by the Department of Automatic Control, Lund Institute of Technology, Sweden. Each kernel represents the interface between the actual dynamical model and the network simulation. Here, continuous simulation and digital conversion take place to transmit information through the network. This tool provides the necessary interruptions to simulate delay propagation as well as synchronization within the network.

4 Concluding Remarks

Present approach shows the integration of three modelling techniques in order to perform reconfiguration. Three approaches are followed, control law design, automaton modelling and RT scheduling strategy. Although there is no formal verification in order to follow this sequence, it has been adopted since structural reconfiguration provides settle conditions for control reconfiguration. The use of a real-time scheduling algorithm in order to approve or disapprove modifications on computer network behaviour allows time delays bounding during a specific time window. This local time delay bounding allows the design of a control law capable to cope with these new conditions. Preliminary results show that control reconfiguration is feasible as long as the use of a switching technique predetermines when one control is the adequate. This goal is reached by a strategy compose of three algorithms, one which is responsible for structural reconfiguration and it has been implemented in this paper as ART2A network. Second

algorithm is responsible for dynamic control design and third algorithm is based on an automaton technique to perform switching control. What it is important for this last approach is that control conditions are strictly bounded to certain response.

Future work is focused to produce certain evaluation metrics that allows feasible comparison between different approaches.

5 Acknowledgements

The authors would like to thank the financial support of DISCA-IIMAS-UNAM, and UNAM-PAPIIT (IN101307 and IN105303) Mexico in connection with this work. And the High Performance Computing Proyect within the "Macroproyecto Tecnologas para la Universidad de la Información y la Computación" of the Universidad Nacional Autónoma de México (UNAM).

Bibliography

[1] Benítez-Pérez H. and García-Nocetti F., Reconfigurable Distributed Control, *Springer Verlag*, 2005.

[2] Blanke M. and Kinnaert M. and Lunze J. and Staroswiecki M., Diagnosis and Fault Tolerant Control, *Springer*, 2003.

[3] Cervin A. and Henriksson D. and Lincoln B. and Eker J. and Arzén K., How Does Control Timing Affect Performance, *IEEE Control Systems Magazine*, Vol. 23, pp. 16-30, 2003.

[4] Gudmundsson D. and Goldberg K., Tuning Robotic Part Feeder Parameters to Maximize Throughput, *Assembly Automation Publisher: MCB University Press*, Vol. 19, No. 3, pp216-221, 1999.

[5] Izadi-Zamanabadi R. and Blanke M., A Ship Propulsion System as a Benchmark for Fault-Tolerant Control, *Control Engineering Practice*, Vol. 7, pp. 227-239, 1999.

[6] Jiang J. and Zhao Q., Reconfigurable Control Based on Imprecise Fault Identification, *Proceedings of the American Control Conference, IEEE*, pp. 114-118, San Diego, June, 1999.

[7] Khalil H., Nonlinear Systems, *Prentice Hall*, 2002.

[8] Nilsson, J., Real-Time Control with Delays, *PhD. Thesis, Department of Automatic Control, Lund Institute of Technology*, Sweden, 1998.

[9] Thompson, H., Wireless and Internet Communications Technologies for monitoring and Control, *Control Engineering Practice*, vol. 12, pp. 781-791, 2004.

[10] Wu N., Reliability of Reconfigurable Control Systems: A Fuzzy Set Theoretic Perspective, *Proceedings of the 36 th Conference on Decision & Control, IEEE*, TP15 5:10, pp. 3352-3356, San-Diego, USA, 1997.

Benítez-Pérez H.
Universidad Nacional Autónoma de México
Departamento de Ingeniería de Sistemas Computacionales y Automatización
Apdo. Postal 20-726., Admón. No. 20
Del. A. Obregón, México D. F., CP. 01000, México.
E-mail: hector@uxdea4.iimas.unam.mx

Cárdenas-Flores F., García-Nocetti, F.
IIMAS, Universidad Nacional Autónoma de México
Apdo. Postal 20-726., Admón. No. 20
Del. A. Obregón, México D. F., CP. 01000, México

Received: August 1, 2007

Hector Benítez-Pérez is a full time researcher in the IIMAS UNAM (México). He obtained his BSc in electronic engineering at the Engineering Faculty UNAM in 1994 and his PhD at Sheffield University, UK en 1999. His areas of interest are in Real Time Control and Fault Diagnosis.

F. Cárdenas-Flores obtained his BSc in Biomedical Engineering at UAM-Iztapalapa 1990. He works in the Departamento de Ingenieria en Sistemas Computacionales IIMAS UNAM since 1994. He areas of interest are in Real time, Parallel computing and Digital Signal and Image Processing.

F. García-Nocetti BSc in electric engeneering at the Engineering Faculty at UNAM in 1984. MBs and PhD at University of Wales, UK 1988, 1991. He is a full time researcher at IIMAS UNAM since 2000. His areas of interest are architectures and algorithms for High performance computing including real time applications. He is member of IEEE, IEE and IFAC.

International Journal of Computers, Communications & Control
Vol. II (2007), No. 4, pp. 328-339

Neural Networks-based Adaptive State Feedback Control of Robot Manipulators

Ghania Debbache, Abdelhak Bennia, Noureddine Goléa

Abstract: This paper proposes an adaptive control suitable for motion control of robot manipulators with structured and unstructured uncertainties. In order to design an adaptive robust controller, with the ability to compensate these uncertainties, we use neural networks (NN) that have the capability to approximate any nonlinear function over a compact space. In the proposed control scheme, we need not derive the linear formulation of robot dynamic equation and tune the parameters. To reduce the NNs complexity, we consider the properties of robot dynamics and the decomposition of the uncertainties terms. The proposed controller is robust against uncertainties and external disturbance. The validity of the control scheme is demonstrated by computer simulations on a two-link robot manipulator.
Keywords: Robot manipulator, neural networks, adaptive control, stability.

1 Introduction

Robot manipulators are multivariable nonlinear systems and are frequently subjected to structured and unstructured uncertainties even in a well-structured setting for industrial use. Structured uncertainties are mainly caused by imprecision in the manipulator link properties, unknown loads, and so on. Unstructured uncertainties are caused by unmodeled dynamics, such as, nonlinear friction, disturbances, and high-frequency dynamics. As a result, it is difficult to obtain an accurate mathematical model so that computed torque controllers [1-6] or other model-based controllers [7-8] can be accurately applied. Although adaptive controllers [1-5, 7] can achieve fine control and compensate for partially unknown manipulator dynamics (i.e., structured uncertainties), they often suffer from incapacity to deal with unstructured uncertainties. Hence, there is a need for model-free adaptive control strategies.

The application of neural networks to robots dynamic control is not new [9-11]. Though the proposed methods have been practically successful, it has proved extremely difficult to develop a general analysis and design theory for early NNs control systems. During the last few years, a number of papers have been presented to deal with the problem of robot adaptive control [12-14]. The basic idea of these methods is to design the feedback controller based on the computed torque principle, and to use an adaptive NN to approximate the robot nonlinearities needed in the control input design. However, most of the above designs present the drawback that, robot dynamic model is presented as single nonlinearity approximated by a single NN with the robot real and desired positions and velocities as inputs, which results in large NN with lot of parameters to be tuned.

In this paper, our goal is to develop a method for designing an adaptive NN control for rigid robot manipulators. A structured or partitioned NN structure, that simplifies the controller design and makes for faster weight tuning online, is designed to ensure the closed loop stability. Robust update laws are used to tune the NNs parameters, and to ensure their boundedness. Lyapunov stability theory is used to drive the stability conditions, and to show the robustness against uncertainties and disturbances. Simulation tests for a two-link robot, under uncertainties, disturbance and parameters variations, show the accuracy and the robustness of the proposed adaptive scheme.

2 Robot control problem

The Lagrange–Euler formulation, the dynamic equation of an n-joint robot arm can be expressed as

$$M(q)\ddot{q}+c(q,\dot{q})+g(q)+\tau_c(q,\dot{q})+\tau_d(q,\dot{q})=u \tag{1}$$

where $M(q)\in R^{n\times n}$ bounded positive definite inertia matrix; $c(q,\dot{q})\in R^n$ vector representing centrifugal and Coriolis effects; $g(q)\in R^n$ vector representing gravitational torques; $\tau_c(q,\dot{q})\in R^n$, $\tau_d(q,\dot{q})\in R^n$ vectors representing the dynamic effects as nonlinear frictions, small joint and link elasticities, backlash and bounded torque disturbances. Here the uncertainties effect is decomposed as continuous part $\tau_c(q,\dot{q})$ and discontinuous part $\tau_d(q,\dot{q})$. $u\in R^n$ vector of joint torques supplied by the actuators; $q\in R^n$ vector of joint positions; $\dot{q}\in R^n$ vector of joint velocities and $\ddot{q}\in R^n$ vector of joint accelerations.

Taking as state vector $x^T=\left[\begin{array}{ccc} x_1^T & \ldots & x_n^T \end{array}\right]$ with $x_i^T=\left[\begin{array}{cc} q_i & \dot{q}_i \end{array}\right]$, the robot model (1) can be rewritten as

$$\dot{x}=Ax+B\left[F(q,\dot{q})+G(q)u+d(q,\dot{q})\right] \tag{2}$$

where

$$F(q,\dot{q})=\left[\begin{array}{c} f_1\left(q,\dot{q}\right) \\ \vdots \\ f_n(q,\dot{q}) \end{array}\right]:=-M^{-1}(q)\left[c\left(q,\dot{q}\right)+g(q)+\tau_c\left(q,\dot{q}\right)\right]$$

$$G(q)=\left[\begin{array}{ccc} g_{11}(q) & \cdots & g_{1n}(q) \\ \vdots & \ddots & \vdots \\ g_{n1}(q) & \cdots & g_{nn}(q) \end{array}\right]:=M^{-1}(q)$$

$$d(q,\dot{q})=\left[\begin{array}{c} d_1\left(q,\dot{q}\right) \\ \vdots \\ d_n(q,\dot{q}) \end{array}\right]:=-M^{-1}(q)\tau_d\left(q,\dot{q}\right)$$

and $A=\text{diag}[A_1,..,A_n]$, $B=\text{diag}[b_1,..,b_n]$ with

$$A_i=\left[\begin{array}{cc} 0 & 1 \\ 0 & 0 \end{array}\right], b_i=\left[\begin{array}{c} 0 \\ 1 \end{array}\right], i=1..n$$

The control problem can be stated as: for a given bounded reference trajectories q_r, $\dot{q}_r$ and $\ddot{q}_r\in R^n$ design the control input torques u such as the robot's states follow their references, with all involved signals in closed loop remain bounded.

3 Neural networks

The general function of one hidden layer feedforward neural network can be described as in (3) as the weighted combination of N activation functions. Here the input vector x and $\varphi_i(.)$ represents the ith activation function (with its parameters vector θ_i) connected to the output by weight w_i.

$$y=\sum_{i=1}^{N}\varphi_i(x,\theta_i)\,w_i \tag{3}$$

The numbers of the input and output layers coincide with the dimension of the input vector and output information number, respectively. Since the above neural networks will be trained on line to achieve the

control task, and in order to reduce computation load, we will assume that the activation functions parameters θ_i are fixed, i.e., their number and shape is a priori determined. The only adjustable parameters are the wights w_i. Then, (3) can be rewritten in the compact form

$$y = w^T \phi(x) \tag{4}$$

where $\phi^T(x) = \left[\begin{array}{ccc} \varphi_1(x) & \ldots & \varphi_N(x) \end{array} \right]$ and $w^T = \left[\begin{array}{ccc} w_1 & \ldots & w_N \end{array} \right]$.

It is known from NNs approximation theory [15-19] that the modeling error can be reduced arbitrarily by increasing the number N, i.e., the number of the linear independent activation functions in the network. That is, a smooth function $f(x)$, $x \in \Omega_x \subset R^n$ can be written as

$$f(x) = w^{*T} \phi(x) + \varepsilon(x) \tag{5}$$

where $\varepsilon(x)$ is the network inherent approximation error, and w^* is an optimal weight vector.

Various well-known results, see e.g. [16-19], for various activation functions $\varphi_i(.)$, based, e.g. on the Stone-Weierstrass theorem, say that any sufficiently smooth function can be approximated by a suitably large NN [5-8]. The functional range of NN (4) is said to be dense, if for any $f(x)$ and a constant $\varepsilon^* > 0$ there exist finite N and w^* such that (5) holds with $|\varepsilon(x)| < \varepsilon^*$. The rang of activation functions include for instance the step, the ramp, the sigmoid and radial basis functions. Several algorithms are proposed in the literature to select the structure and parameters for those kind of NNs, see e.g. [20-21].

4 Neural state feedback

Due to approximation property (5), we can assume that the nonlinear terms in (2) can be approximated as

$$\begin{array}{c} f_i(q,\dot{q}) = \theta_{f_i}^{*T} \phi_i(q,\dot{q}) + \varepsilon_i(q,\dot{q}) \\ g_{ii}(q) = \theta_{g_i}^{*T} \psi_i(q) + \varepsilon_i(q) \end{array} \quad i = 1..n \tag{6}$$

where $\theta_{f_i}^{*T} \phi_i(q,\dot{q})$ and $\theta_{g_i}^{*T} \psi_i(q)$ are NNs of the from (4), and $\varepsilon_i(q,\dot{q})$, $\varepsilon_i(q)$ are the inherent approximation errors due to the finite size of the NNs. The optimal weights $\theta_{f_i}^*$ and $\theta_{g_i}^*$ defined above are quantities required only for analytical purpose. Typically $\theta_{f_i}^*$ and $\theta_{g_i}^*$ are chosen to minimize $\varepsilon_i(q,\dot{q})$ and $\varepsilon_i(q)$ over the compact regions Ω_f and Ω_g respectively, that is

$$\theta_{f_i}^* = \arg\min_{\theta_{f_i}} \left\{ \sup_{q,\dot{q} \in \Omega_f} \left| f_i(q,\dot{q}) - \theta_{f_i}^T \phi_i(q,\dot{q}) \right| \right\}$$

$$\theta_{g_i}^* = \arg\min_{\theta_{g_i}} \left\{ \sup_{q \in \Omega_g} \left| g_{ii}(q) - \theta_{g_i}^T \psi_i(q) \right| \right\}$$

Assumption 1: The neural networks approximation errors are bounded by $\left|\varepsilon_i(q,\dot{q})\right| \leq \varepsilon_{0i}$ and $|\varepsilon_i(q)| \leq \varepsilon_{0i}$, $i = 1..n$, for some constants ε_{0i} and ε_{0i}.

Assumption 1 results from the universal approximation property of neural networks, that can approximate any well-defined function over a compact space with finite approximation error.

Using (6) in (2), the robot dynamic can be written as

$$\dot{x} = Ax + B\left[\Theta_f^* \Phi(q,\dot{q}) + \Theta_g^* \Psi(q) u + H(q)u + \omega(q,\dot{q})\right] \tag{7}$$

where $\Phi(q,\dot{q})$ =block-diag$\left[\phi_1(q,\dot{q}),..,\phi_n(q,\dot{q})\right]$, $\Psi(q,\dot{q})$ =block-diag$\left[\psi_1(q,\dot{q}),..,\psi_n(q,\dot{q})\right]$, Θ_f^* =block-

$\text{diag}\left[\theta_{f_1}^{*T},..,\theta_{f_n}^{*T}\right]$, Θ_g^* =block-diag$\left[\theta_{g_1}^{*T},..,\theta_{g_n}^{*T}\right]$, $\omega(q,\dot{q}) = \varepsilon + d(q,\dot{q})$, with $\varepsilon^T = \begin{bmatrix} \varepsilon_1 & ... & \varepsilon_n \end{bmatrix}$, and

$$H(q) = \begin{bmatrix} \varepsilon_1 & g_{12}(q) & \cdots & g_{1n}(q) \\ g_{21}(q) & \ddots & & \vdots \\ \vdots & & \ddots & g_{(n-1)n}(q) \\ g_{n1}(q) & \cdots & g_{n(n-1)}(q) & \varepsilon_n \end{bmatrix}$$

Based on (7), the control inputs are defined as

$$u = [\Theta_g \Psi(q)]^{-1}\left[-\Theta_f \Phi(q,\dot{q}) + \ddot{q}_r + Ke\right] \tag{8}$$

where $e^T = \begin{bmatrix} (q_r - q)^T & \left(\dot{q}_r - \dot{q}\right)^T \end{bmatrix}$ is the tracking error vector, Θ_g, Θ_f are the estimated neural networks parameters, and K =diag$[K_1,...,K_n]$ with $K_i \in R^2$ is PD gain vector, chosen such as the matrix $A_c = A - BK$ is Hurwitz.

Then, introducing the control input (8) in (7) yields

$$\dot{e} = A_c e - B\left[\widetilde{\Theta}_f \Phi(q,\dot{q}) + \widetilde{\Theta}_g \Psi(q) u + H(q)u + \omega(q,\dot{q})\right] \tag{9}$$

where $\widetilde{\Theta}_f = \Theta_f^* - \Theta_f$ and $\widetilde{\Theta}_g = \Theta_g^* - \Theta_g$ are the parameters estimation errors.

From (9), it can be seen that the tracking error vector is driven by the coupling terms and the finite approximation accuracy effects reflected by $H(q)$ and the uncertainty term $\omega(q,\dot{q})$.

To design the neural networks parameters update laws and to ensure boundedness of the involved signals in the closed loop robot control, the following assumptions are used:

Assumption 2: The diagonal elements of $G(q)$ are bounded such as $g_m \leq$diag$[g_{11}(q),..,g_{nn}(q)] \leq g_M$, the matrix $H(q)$ is bounded by $|H(q)| \leq h_0$, and the disturbance term $\omega(q,\dot{q})$ is bounded by $\left|\omega(q,\dot{q})\right| \leq \omega_0$.

Assumption 3: The neural networks parameters are bounded by the constraint sets Ω_f and Ω_g such that: $\Omega_f = \left\{\Theta_f \mid \left|\Theta_f\right| \leq f_M\right\}$ and $\Omega_g = \left\{\Theta_g \mid g_m \leq |\Theta_g| \leq g_M\right\}$, respectively, where f_M, g_m, and g_M are some known constants.

The first part of assumption 2 follows from the fact that $M(q)$ is bounded positive definite matrix, the second part follows from the boundedness of $M(q)$ and $\varepsilon_i(q)$. Finally, the third part follows from boundedness of $M(q)$, τ_d and $\varepsilon_i(q,\dot{q})$. The bounds used in assumption 3 result from the assumptions 1-2 and are used to ensure the boundedness of the neural networks outputs.

In order to constraint the parameters Θ_f and Θ_g within the sets Ω_f and Ω_g, respectively, we use the following parameter projection algorithm [22]:

$$\dot{\Theta}_f = \begin{cases} -\gamma_1 B^T Pe\Phi^T(q,\dot{q}) & \text{if } \left|\Theta_f\right| < f_M \text{ or } (\left|\Theta_f\right| = f_M \\ & \text{and tr}\left[B^T Pe\Phi^T(q,\dot{q})\Theta_f^T\right] \geq 0) \\ -\gamma_1 B^T Pe\Phi^T(q,\dot{q}) & \text{if } \left|\Theta_f\right| = f_M \text{ and} \\ +\gamma_1 \text{tr}\left[B^T Pe\Phi^T(q,\dot{q})\widehat{\Theta}_f^T\right]\left(\frac{1+|\Theta_f|}{f_M}\right)^2\Theta_f & \text{tr}\left[B^T Pe\Phi^T(q,\dot{q})\Theta_f^T\right] < 0 \end{cases} \tag{10}$$

and

$$\dot{\Theta}_g = \begin{cases} -\gamma_2 B^T Peu^T\Psi^T(q) & \text{if } \left|\widehat{\Theta}_g\right| < g_M \text{ or } (\left|\widehat{\Theta}_g\right| = g_M \\ & \text{and tr}\left[B^T Peu^T\Psi^T(q)\widehat{\Theta}_g^T\right] \geq 0) \\ -\gamma_2 B^T Peu^T\Psi^T(q) & \text{if } |\Theta_g| = g_M \text{ and} \\ +\gamma_2 \text{tr}\left[B^T Peu^T\Psi^T(q)\Theta_g^T\right]\left(\frac{1+|\Theta_g|}{g_M}\right)^2\Theta_g & \text{tr}\left[B^T Peu^T\Psi^T(q)\Theta_g^T\right] \leq 0 \end{cases} \tag{11}$$

where $\gamma_1, \gamma_2 > 0$ are design parameters, and $P = P^T > 0$ is the solution, for a given $Q = Q^T > 0$, of the Lyapunov equation

$$A_c^T P + P A_c = -Q \tag{12}$$

Moreover, in order to guarantee $|\Theta_g| \geq g_m$ such that an inverse of $\Theta_g \Psi(q)$ always exists, we use the following law to adjust the parameter Θ_g.

1. Whenever any element $[\Theta_g]_{ij} = g_m$ use

$$\left[\dot{\Theta}_g\right]_{ij} = \begin{cases} -\gamma_2 \left[B^T P e u^T \Psi^T(q)\right]_{ij} & \text{if } \left[B^T P e u^T \Psi^T(q)\right]_{ij} < 0 \\ 0 & \text{if } \left[B^T P e u^T \Psi^T(q)\right]_{ij} \geq 0 \end{cases} \tag{13}$$

2. Otherwise, use (11).

where $[A]_{ij}$ stands for the ijth element of the matrix A.

The stability properties of the proposed NNs adaptive state feedback are summarized by the following theorem.

Theorem 1: The robot adaptive control composed by the robot dynamic (2), the control input (8), the update laws (10)-(11) and (13) verifying assumptions 1-3, guarantees the following:

1. $\left|\Theta_f\right| \leq f_M$ and $g_m \leq |\Theta_g| \leq g_M$
2. $|e| \in L_\infty$
3. $|u| \in L_\infty$

Proof:

1. To prove $|\Theta_g| \leq g_M$, let $V_g = \frac{1}{2\gamma_2} \mathrm{tr}\left[\Theta_g^T \Theta_g\right]$, then $\dot{V}_g = \frac{1}{\gamma_2} \mathrm{tr}\left[\dot{\Theta}_g^T \Theta_g\right]$. If the first line of (11) is true, we have either $|\Theta_g| < g_M$ or $\dot{V}_g = -\mathrm{tr}\left[\left(B^T P e u^T \Psi^T(q)\right)^T \Theta_g\right] \leq 0$ when $|\Theta_g| = g_M$, that is, we get always $|\Theta_g| \leq g_M$. If the second line of (11) is true, we have $|\Theta_g| = g_M$ and

$$\begin{aligned} \dot{V}_g &= \mathrm{tr}\left[-\left(B^T P e u^T \Psi^T(q)\right)^T \Theta_g + \mathrm{tr}\left[B^T P e u^T \Psi^T(q) \Theta_g^T\right] \left(\frac{1+|\Theta_g|}{g_M}\right)^2 \Theta_g^T \Theta_g\right] \\ &= -\mathrm{tr}\left[\left(B^T P e u^T \Psi^T(q)\right)^T \Theta_g\right] + \mathrm{tr}\left[B^T P e u^T \Psi^T(q) \Theta_g^T\right] \left(\frac{1+|\Theta_g|}{g_M}\right)^2 \mathrm{tr}\left[\Theta_g^T \Theta_g\right] \\ &= -\mathrm{tr}\left[\left(B^T P e u^T \Psi^T(q)\right)^T \Theta_g\right] + \mathrm{tr}\left[B^T P e u^T \Psi^T(q) \Theta_g^T\right] \left(\frac{1+|\Theta_g|}{g_M}\right) |\Theta_g|^2 \end{aligned} \tag{14}$$

since $|\Theta_g| = g_M$, we get

$$\dot{V}_g = \mathrm{tr}\left[B^T P e u^T \Psi^T(q) \Theta_g^T\right] g_M^2 \leq 0 \tag{15}$$

that is, $|\Theta_g| \leq g_M$. Therefore, we have $|\Theta_g| \leq g_M$, $\forall t \geq 0$.

From (13) we see that if $|\Theta_g|_{ij} = g_m$, then $\left[\dot{\Theta}_g\right]_{ij} \geq 0$; that is, we have that $|\Theta_g|_{ij} \geq g_m$.

Using the same analysis, we can prove that $\left|\Theta_f\right| \leq f_M$, $\forall t \geq 0$.

2. Consider the Lyapunov function

$$V = \frac{1}{2}e^T Pe + \frac{1}{2\gamma_1}\text{tr}\left[\widetilde{\Theta}_f^T\widetilde{\Theta}_f\right] + \frac{1}{2\gamma_2}\text{tr}\left[\widetilde{\Theta}_g^T\widetilde{\Theta}_g\right] \tag{16}$$

The differentiation of (16) along (9) yields

$$\begin{aligned}\dot{V} = &-\frac{1}{2}e^T Qe - e^T PB\left[\widetilde{\Theta}_f\Phi(q,\dot{q}) + \widetilde{\Theta}_g\Psi(q)u + H(q)u + \omega(q,\dot{q})\right] \\ &+ \frac{1}{\gamma_1}\text{tr}\left[\dot{\widetilde{\Theta}}_f^T\widetilde{\Theta}_f\right] + \frac{1}{\gamma_2}\text{tr}\left[\dot{\widetilde{\Theta}}_g^T\widetilde{\Theta}_g\right]\end{aligned} \tag{17}$$

which can be arranged as

$$\begin{aligned}\dot{V} = &-\frac{1}{2}e^T Qe - e^T PB\left[H(q)u + \omega(q,\dot{q})\right] \\ &+ \frac{1}{\gamma_1}\text{tr}\left[\left(\dot{\widetilde{\Theta}}_f^T - \gamma_1\Phi(q,\dot{q})e^T PB\right)\widetilde{\Theta}_f\right] + \frac{1}{\gamma_2}\text{tr}\left[\left(\dot{\widetilde{\Theta}}_g^T - \gamma_2\Psi(q)ue^T PB\right)\widetilde{\Theta}_g\right]\end{aligned} \tag{18}$$

Then, using (10)-(11) and the fact that $\dot{\widetilde{\Theta}}_f = -\dot{\Theta}_f$ ($\dot{\widetilde{\Theta}}_g = -\dot{\Theta}_g$), one can show that the third and fourth terms in (18) are always ≤ 0. Therefore, (18) can be written as

$$\dot{V} \leq -\frac{1}{2}e^T Qe - e^T PB\left[H(q)u + \omega(q,\dot{q})\right] \tag{19}$$

Further, (19) can be upper bounded by

$$\dot{V} \leq -\frac{1}{2}\lambda_{\min}(Q)|e|^2 + |e||PB|\left[|H(q)||u| + \left|\omega(q,\dot{q})\right|\right] \tag{20}$$

Then, using (8) and the result in 1, the input torques can be upper bounded by

$$\begin{aligned}|u| &\leq \left|[\Theta_g\Psi(q)]^{-1}\right|\left[\left|\Theta_f\Phi(q,\dot{q})\right| + \left|\ddot{q}_r\right| + |K||e|\right] \\ &\leq \frac{1}{g_m}\left(f_M + q_0 + |K||e|\right)\end{aligned} \tag{21}$$

where q_0 is an upper bound on the desired accelerations $\ddot{q}_r$.

Then, using (21) and the assumptions 1-3 in (20), becomes

$$\dot{V} \leq -\frac{1}{2}\lambda_{\min}(Q)|e|^2 + \kappa_1|e|^2 + \kappa_2|e| \tag{22}$$

where $\kappa_1 = \frac{h_0}{g_m}|PBK|$ and $\kappa_2 = |PB|\left(\frac{h_0}{g_m}(f_M + q_0) + \omega_0\right)$.

Hence, (22) can be arranged as

$$\dot{V} \leq -\frac{1}{2}\left(\lambda_{\min}(Q) - 2\kappa_1\right)|e|^2 + \kappa_2|e| \tag{23}$$

If the matrix Q is chosen such as $\lambda_{\min}(Q) > 2\kappa_1$, then $\dot{V} \leq 0$ whenever the tracking error is outside the region given by

$$|e| \leq \frac{2\kappa_2}{\left(\lambda_{\min}(Q) - 2\kappa_1\right)} \tag{24}$$

which implies that $|e| \in L_\infty$.

3. Using the result (24) in (21) yields

$$|u| \leq \frac{1}{g_m}\left(f_M + q_0 + |K|\,\frac{2\kappa_2}{(\lambda_{\min}(Q) - 2\kappa_1)}\right) \tag{25}$$

this implies that $|u| \in L_\infty$.

Remarks:

1. Only the diagonal elements of $G(q)$ are estimated and used in the control inputs design. By doing this, we avoid the estimation of the coupling terms (considered here as disturbances) and the need to compute the inverse of the estimation of $G(q)$.
2. Although, the control torques (8) are presented in vector form, they can be, in practice, computed independently, since $\Theta_g \Psi(q)$ and K are diagonal matrices and no information is needed from the other input torques.
3. From (25), it can be seen that constraints on the control inputs, i.e., $|u_i| \leq u_{i\max}$ can be meet by tuning the the PD gain K_i and the desired accelerations magnitudes q_0.

5 Simulation Results

To test the proposed adaptive neural control, we consider the two-link manipulator (fig. 1) whose dynamic equations of motion (1) are:

$$M(q) = \begin{bmatrix} (m_1+m_2)\,l_1^2 & m_2 l_1 l_2\,(s_1 s_2 + c_1 c_2) \\ m_2 l_1 l_2\,(s_1 s_2 + c_1 c_2) & m_2 l_2^2 \end{bmatrix}$$

$$c\left(q,\dot{q}\right) = \begin{bmatrix} 0 & -m_2 l_1 l_2\,(c_1 s_2 - s_1 c_2)\,\dot{q}_2 \\ -m_2 l_1 l_2\,(c_1 s_2 - s_1 c_2)\,\dot{q}_1 & 0 \end{bmatrix}$$

$$g(q) = \begin{bmatrix} -g\,(m_1+m_2)\,s_1 l_1 \\ -g m_2 l_2 s_2 \end{bmatrix}$$

where $c_1 = \cos(q_1)$, $c_2 = \cos(q_2)$, $s_1 = \sin(q_1)$ and $s_2 = \sin(q_2)$. The robot physical parameters are: $l_1 = l_2 = 1\text{m}$, $m_1 = m_2 = 1\text{kg}$, and $g = 9.81\ \text{m/s}^2$.

The uncertainties terms in (1) are given by:

$$\tau_c = \begin{bmatrix} \dot{q}_1 + \sin(3q_1) \\ 1.2\dot{q}_2 + 0.5\sin(2q_2) \end{bmatrix},\ \tau_d = \begin{bmatrix} 0.2\text{sign}\left(\dot{q}_1\right) \\ 0.1\text{sign}\left(\dot{q}_2\right) \end{bmatrix}$$

The NNs adaptive control design for the two-link robot is as follows:

1. In order to construct the NNs approximators, each variable $q_1, q_2 \in [-\pi, \pi]$ and $\dot{q}_1, \dot{q}_2 \in [-2\pi, 2\pi]$ range is devised into 3 sub domains, which yields four RBF networks to approximate $f_1\left(q,\dot{q}_2\right), f_2\left(q,\dot{q}_1\right), g_{11}(q)$ and $g_{22}(q)$, with $q^T = \begin{bmatrix} q_1 & q_2 \end{bmatrix}$, with, respectively, 27, 27, 9 and 9 RBFs. The designed RBF networks take the following compact forms:

$$\widehat{f}_1\left(q,\dot{q}_2\right) = \theta_{f_1}^T \phi_1\left(q,\dot{q}_2\right) \tag{26}$$

$$\widehat{f}_2\left(q,\dot{q}_1\right) = \theta_{f_2}^T \phi_2\left(q,\dot{q}_1\right) \tag{27}$$

$$\widehat{g}_{11}(q) = \theta_{g_1}^T \psi_1(q) \tag{28}$$

$$\widehat{g}_{22}(q) = \theta_{g_2}^T \psi_2(q) \tag{29}$$

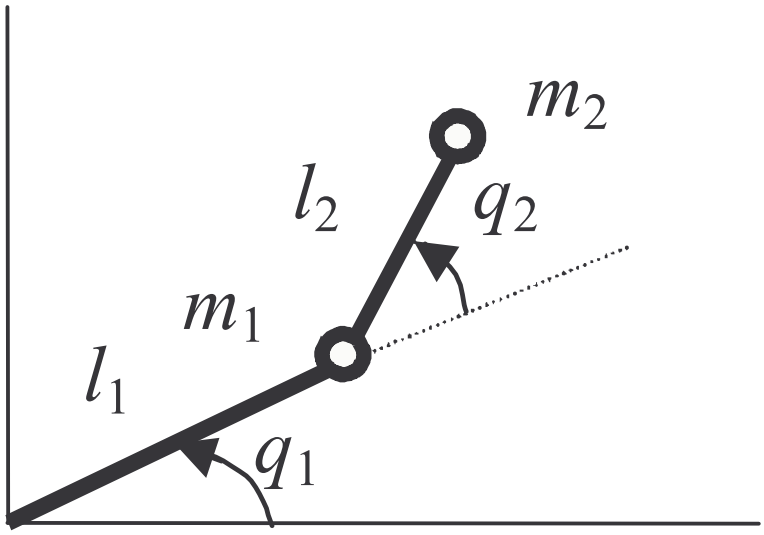


Figure 1: Two link robot

where $\theta_{f_1}, \theta_{f_2} \in R^{27\times 1}$ and $\theta_{g_1}, \theta_{g_2} \in R^{9\times 1}$ are the NNs adjustable parameters. This approach is far from being optimal, but it has the merit to reduce the number of parameters to be learned and thus to reduce the update algorithm complexity and execution time.

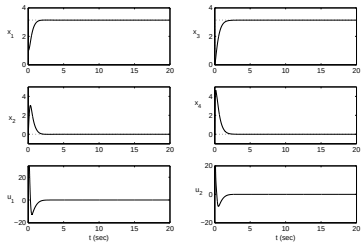
2. The control input is designed as in (8) with the adaptive NNs defined in (26)-(29). The PD gain is defined as $K = \text{diag}[K_1, K_2]$ with $K_1 = K_2 = \begin{bmatrix} 16 & 8 \end{bmatrix}$.

3. For the choice of $Q = 100I_4$ and the solutions of (12) we get P.

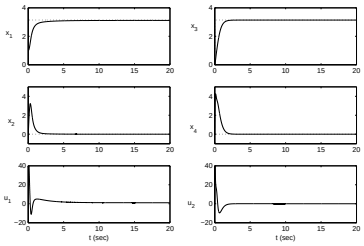
4. By analyzing the dynamic of the robot, the following bounds are fixed $g_M = 2.5$, $g_m = 0.5$ and $f_M = 20$. The adjustable parameters are updated using (10)-(11) with $\gamma_1 = 0.01$ and $\gamma_2 = 0.001$.

In the simulation, the NNs parameters are initialized as $|\theta_{g_1}(0)| = |\theta_{g_2}(0)| = 0.5$ and $\left|\theta_{f_1}(0)\right| = \left|\theta_{f_2}(0)\right| = 0$. The initial states, in all simulations, are $x^T(0) = \begin{bmatrix} 0.5 & 2 & -0.5 & 1 \end{bmatrix}$.

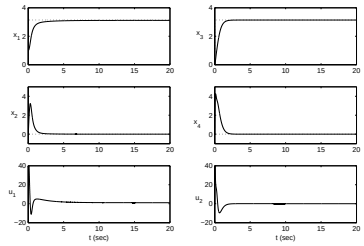
The first simulation test concerns the regulation of the joint positions under nominal conditions, i.e., no parameters changes and no disturbances. As depicted in figure 2.a, the joint positions exhibit a good transient performance, and no error is remarked in steady state regime. Figure 2.b shows the regulation performance under uncertainties effects. From this figure it is seen that these uncertainties affect little the regulation performance and small steady state error is introduced and the NNs adaptive control achieves good compensation of the uncertainties effects. In figure 2.c, in addition to the uncertainties effects, a payload change is introduced at $t = 15$s when m_2 passes from 1kg to 3kg. This situation is rapidly taken in account by the NNs control and it's effect is compensated.



a) nominal case

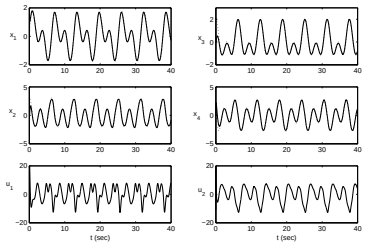


b) $\tau_c + \tau_d$ uncertainties effects

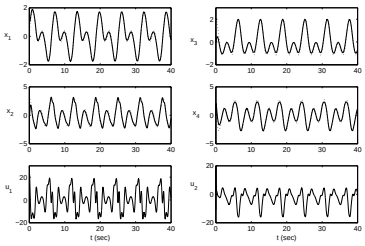


c) $\tau_c + \tau_d + m_2$ variation effects

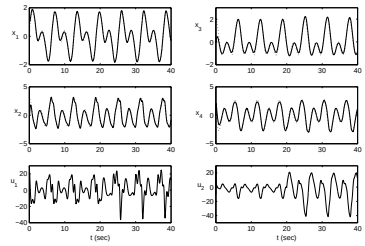
Figure 2: Regulation performance



a) nominal case



b) $\tau_c + \tau_d$ uncertainties effects



c) $\tau_c + \tau_d + m_2$ variation effects

Figure 3: Tracking performance

The second simulation test concerns the tracking performance for the desired trajectories $q_{r1} = \sin(t) + \sin(2t)$ and $q_{r2} = \cos(t) + \cos(2t)$. Figure 3.a presents the tracking performance in the nominal case. As depicted, the joint positions exhibit a good transient performance, and small error is remarked in steady state regime. Figure 3.b shows the tracking performance under uncertainties effects. It is clear that these uncertainties introduce acceptable tracking error, and the NNs control inputs compensate the uncertainties with a little effort. In figure 3.c, we add to the uncertainties effects, a payload change is introduced at $t = 20$s when m_2 passes from 1kg to 3kg. This variation affects essentially the developed torques to compensate the additional mass, and small error is remarked.

6 Conclusion

In this paper, an adaptive NNs control scheme for rigid robot control, was proposed. The adaptive capability of handling modeling errors and external disturbances was demonstrated. The error convergence rate with the NNs adaptive approach was found to be fast. Asymptotic stability of the control system is established using the Lyapunov approach. Simulation studies for a two-link robot verify the flexibility, adaptation and tracking performance of the proposed approach. The major contributions of the paper are as follows: reduction of the NNs complexity and no robustifying control is required to achieve the stability or to enhance the control performance.

Bibliography

[1] J. Y. Han, H. Hemami, and S. Yurkovich, Nonlinear adaptive control of a N-link robot with unknown load, Int. J. Robotics Res., vol. 6, no. 3, pp. 71–86, 1987.

[2] J. E. Slotine and W. Li, On the adaptive control of robot manipulators, Int. J. Robotics Res., vol. 6, no. 3, pp. 49–59, 1987.

[3] R. Ortega and M. Spong, Adaptive motion control of rigid robots: A tutorial, Proc. 27th Conf. on Decision and Control CDC, Austin, Texas, , pp. 1575-1584, Dec 1988.

[4] J. J. Graig, Adaptive Control of Mechanical Manipulators, New York, Addison-Wesley, 1988.

[5] Z. Qu, J. F. Dorsey and D. M. Dawson, Robust Control of robots by computed torque law, Systems and Control Letter, Vol.16, pp.25-32, 1991.

[6] G. Jumarie, Tracking control of mechanical systems via sliding Lagrangian, J. Intell. Robotic Syst., vol. 13, pp. 181–199, 1995.

[7] S. Pandian and M. Hanmandlu, Adaptive generalized model-based control of robot manipulators, Int. J. Contr., vol. 58, no. 4, pp. 835-853, 1993.

[8] W. C. Ham, Adaptive control based on explicit model of robot manipulators, IEEE Trans. Automat. Contr., vol. 38, pp. 654–658, 1993.

[9] M. B. Leahy, M. A. Johnson, S. K. Rogers, Neural network payload estimation for adaptive robot control, IEEE Transactions on Neural Networks, Vol. 2, No. 1, pp. 93-100, Jan. 1991.

[10] M. Saad, P. Bigras, L.-A. Dessaint, K. Al-Haddad, Adaptive robot control using neural networks, IEEE Transactions on Industrial Electronics, Vol. 41, No. 2, pp. 173-181, Apr 1994.

[11] L. Behera, M.Gopal and S.Chaudhury, Inversion of RBF networks and applications to adaptive control of nonlinear systems, IEE Proc.-Control Theory Appl., Vol. 142, No. 6, Nov. 1995.

[12] F. L. Lewis, A. Yesildirak , S. Jagannathan, Neural network control of robot manipulators and nonlinear systems, Taylor & Francis, Inc. Bristol, PA, USA 1998.

[13] M. Quoy, S. Moga, P. Gaussier, Dynamical neural networks for planning and low-level robot control, IEEE Transactions on Systems, Man and Cybernetics, Part A, Vol. 33, No. 4, pp. 523-532, July 2003.

[14] C. Torras, Natural inspiration for artificial adaptivity: some neurocomputing experiences in robotics, Proc. 4th Intl. Conf. on Unconventional Computation (UC'05), Sevilla, Spain, Oct. 2005.

[15] M. J. D. Powell, "Radial basis functions for multivariable interpolation: A review," in *Algorithms for Approximation of Functions and Data*, J. C. Mason, M. G. Cox, Eds. Oxford, U.K.: Oxford Univ. Press, 1987, pp. 143-167.

[16] E. J. Hartman, J. D. Keeler, and J. M. Kowalski, "Layered neural networks with gaussian hidden units as universal approximations," *Neural Comp.*, vol. 2, no. 2, pp. 210-215, 1990.

[17] T. Poggio and F. Girosi, "Networks for approximation and learning," *Proc. IEEE*, vol. 78, pp. 1481-1497, Sept. 1990.

[18] J. Park and I. W. Sandberg, Universal approximation using radial-basis-function networks, Neural Comp. 3, 246-257, 1991.

[19] B. Liu and J. Si, The best approximation to C^2 function and its error bounds using regular-center gaussian networks, IEEE Trans. Neural Networks, vol. 5, pp.848-857, 1994.

[20] R.J. Schilling, J.J. Jr. Carroll, A.F Al-Ajlouni, Approximation of nonlinear systems with radial basis function neural networks, IEEE Transactions on Neural Networks, vol. 12, pp. 1-15, 2001.

[21] H. Guang-Bin, P. Saratchandran, N. Sundararajan, A generalized growing and pruning RBF (GGAP-RBF) neural network for function approximation, IEEE Transactions on Neural Networks, vol. 16, pp. 57- 67, 2005

[22] S. S. Sastry and M. Bosdon, Adaptive control: Stability, Convergence and Robustness, Printice-Hall, 1989.

Ghania Debbache
Electrical Engineering Institute, Oum El-Bouaghi University, 04000 Oum El-Bouaghi, Algeria
gdebbache@yahoo.fr

Abdelhak Bennia
Electronic Departement, Constantine University
25000 Constantine, Algeria
abdelhak.bennia@laposte.net

Noureddine Goléa
Electrical Engineering Institute, Oum El-Bouaghi University, 04000 Oum El-Bouaghi, Algeria
n.golea@lycos.com

Received: March 15, 2007

Ghania Debbache was born in Constantine, Algeria, in 1974. She received the B.Sc. and M.Sc. degrees in electronics from the Constantine University, Algeria, in 1997 and 2000, respectively. She is currently pursuing the Ph.D. degree on the intelligent control at the University Constantine. Currently, she is a Teaching Assistant at the Technologic Sciences Institute of Oum El Bouaghi University, Algeria.

Abdelhak Bennia received his D.E.S. degree in 1983 in physics from the University of Constantine, Algeria. From 1984 to 1990 he attended the graduate school at Virginia Tech, majoring in electrical engineering. He received the M.Sc. degree in 1986 and the Ph.D. degree in 1990. Since 1990, he has been with the electronics department of the University of Constantine. His current research interests are deconvolution, system identification and neural networks applied to character recognition, control systems, and signal processing.

Noureddine Goléa was born in Batna, Algeria, in 1967. He received the Engineer and Master grades from Sétif University, Algeria, and the Doctorat from Batna University, Algeria, all in industrial control, in 1991, 1994, and 2000, respectively. From 1991 to 1994, he was with Electronics Institute at Sétif University, and from 1994 to 1996, he was with the Electronics Institute at Batna University. Currently, he is Professor of electrical engineering in the Technologic Sciences Institute at Oum El-Bouaghi University, Algeria. His research interests are nonlinear and adaptive control, and intelligent control applied to motion control.

International Journal of Computers, Communications & Control
Vol. II (2007), No. 4, pp. 340-354

Bounded Controllers for Decentralized Formation Control of Mobile Robots with Limited Sensing

K.D. Do

Abstract: This paper presents a constructive method to design bounded cooperative controllers that force a group of N mobile robots with limited sensing ranges to stabilize at a desired location, and guarantee no collisions between the robots. The control development is based on new general potential functions, which attain the minimum value when the desired formation is achieved, and are equal to infinity when a collision between any robots occurs. Smooth and p times differential jump functions are introduced and embedded into the potential functions to deal with the robot limited sensing ranges. Formation tracking is also considered.
Keywords: Formation control, mobile robot, local potential function, nonholonomic mobile robot.

1 Introduction

Formation is an extremely useful tool mimicking from biological systems to man-made teams of vehicles, mobile sensors and embedded robotic systems to perform tasks such as jointly moving in a synchronized manner or deploying over a given region with applications to search, rescue, coverage, surveillance, reconnaissance and cooperative transportation. Formation control can be roughly understood as controlling positions of a group of the robots such that they stabilize/track desired locations relative to reference point(s), which can be another robot(s) within the team, and can either be stationary or moving. Three popular approaches to formation control are leader-following (e.g. [2], [3]), behavioral (e.g. [4], [5]), and use of virtual structures (e.g. [6], [7]. Most research works investigating formation control utilize one or more of these approaches in either a centralized or decentralized manner. Centralized control schemes, see e.g. [3], [11], and [10] where actual dynamics of nonholonomic robots were considered in the formation control design, use a single controller that generates collision free trajectories in the workspace. Although these guarantee a complete solution, centralized schemes require high computational power and are not robust due to the heavy dependence on a single controller. A nice application of formation control based on potential field method [3] and Lyapunov's direct method [17] to gradient climbing is recently addressed in [18]. However, the final configuration of formation cannot be foretold. On the other hand, decentralized schemes, see e.g. [8], [12], require less computational effort, and are relatively more scalable to the team size. The decentralized approach usually involves a combination of robot based local potential fields (e.g. [3], [15], [16]). The main problem with the decentralized approach, when collision avoidance is taken into account, is that it is extremely difficult to predict and control the critical points of the controlled systems. Recently, a method based on a different navigation function from [13] provided a centralized formation stabilization control design strategy is proposed in [11]. This work is extended to a decentralized version in [12]. However, the potential function, which possesses all properties of a navigation function (see [13]), is finite (but its gradient with respect to the system states can be unbounded) when a collision occurs. This complicates analysis of collision avoidance. Moreover, the formation is stabilized to any point in workspace instead of being "tied" to a fixed coordinate frame. In [13], [11] and [12], the tuning constants, which are crucial to guarantee that the only desired equilibrium points are asymptotic stable and that the other critical points are unstable, are extremely difficult to obtain. This problem has been removed in [9] where new potential functions were introduced. It is however noted that in [9] each robot requires knowledge of position of all other robots in the group. Moreover, the control design methods (e.g. [3], [14], [15], [18]) based on the potential functions that are equal to infinity when a collision occurs exhibit very large control efforts if the robots

are close to each other. Hence, a bounded control is called for. In addition, switching control theory [21] is often used to design a decentralized formation control system (e.g. [2], where a case by case basis is proposed), especially when the vehicles have limited sensing ranges and collision avoidance between vehicles must be considered. Clearly, it is more desirable if we are able to design a non-switching formation control system that can handle the above decentralized and collision avoidance requirements.

In this paper, bounded cooperative controllers are designed for formation stabilization of a group of mobile robots with limited sensing ranges. New general potential functions are constructed to design the controllers that yield (almost) global asymptotic convergence of a group of mobile robots to a desired formation, and guarantee no collisions among the robots. Smooth and p times differential jump functions are introduced and embedded into the potential functions to deal with the robot limited sensing ranges. Moreover, the controlled system exhibits multiple equilibria due to collision avoidance taken into account. We therefore investigate the behavior of equilibrium points by linearizing the closed loop system around those points, and show that critical points, other than the desired point for an robot, are unstable. The proposed formation stabilization solution is then extended to solve a formation tracking problem.

2 Problem statement

We consider a group of N mobile robots, of which each has the following dynamics

$$\dot{q}_i = u_i, i = 1, ..., N \tag{1}$$

where $q_i \in \mathbb{R}^n$ and $u_i \in D \subset \mathbb{R}^n$ are the state and control input of the robot i. We assume that $n > 1$ and $N > 1$. Here, we treat each robot as an autonomous point.

Control objective: Assume that at the initial time $t_0 \geq 0$ each robot starts at a different location, and that each robot has a different desired location, i.e. there exist strictly positive constants ε_1 and ε_2 such that for all $(i, j) \in \{1, 2, ..., N\}, i \neq j$

$$\|q_i(t_0) - q_j(t_0)\| \geq \varepsilon_1, \|q_{if} - q_{jf}\| \geq \varepsilon_2 \tag{2}$$

where $q_{if}, i = 1, .., N$, is the desired location of the robot i. Moreover, the robot i can only measure its own state and can only detect the other group members if these members are in a sphere, which is centered at the robot and has a radius of R_i larger than a strictly positive constant. Design the bounded control input u_i for each robot i such that each robot asymptotically approaches its desired location while avoids collisions with all other robots in the group, i.e. for all $(i, j) \in \{1, 2, ..., N\}, i \neq j, t \geq t_0 \geq 0$

$$\|u_i(t)\| \leq \delta, \lim_{t \to \infty}(q_i(t) - q_{if}) = 0, \|q_i(t) - q_j(t)\| \geq \varepsilon_3 \tag{3}$$

where δ is a strictly positive constant and ε_3 is a positive constant.

3 Preliminaries

We present one definition and one lemma to be used in the control design and stability analysis in the next section.

Definition 1. A scalar function $h(x,a,b)$ is called a p times differential jump function if it enjoys the properties:

$$\begin{aligned}
&1)\ h(x,a,b)=0 \quad \text{if} \quad 0\le x\le a,\\
&2)\ h(x,a,b)=1 \quad \text{if} \quad x\ge b,\\
&3)\ 0<h(x,a,b)<1 \ \text{if}\ a<x<b,\\
&4)\ h(x,a,b)\ \text{is}\ p\ \text{times differentiable with respect to}\ x
\end{aligned} \tag{4}$$

where p is a positive integer, $x\in\mathbb{R}_+$, and a and b are constants such that $0\le a<b$. Moreover, if $p=\infty$ then the function $h(x,a,b)$ is called a smooth jump function.

Lemma 1. Let the scalar function $h(x,a,b)$ be defined as

$$h(x,a,b)=\frac{\int_a^x f(\tau-a)f(b-\tau)d\tau}{\int_a^b f(\tau-a)f(b-\tau)d\tau} \tag{5}$$

with the function $f(y)$ being defined as follows

$$f(y)=0 \text{ if } y\le 0, \quad f(y)=g(y) \text{ if } y>0 \tag{6}$$

where the function $g(y)$ enjoys the following properties

$$\begin{aligned}
&a)\quad g(\tau-a)g(b-\tau)>0 \quad a<\tau<b,\\
&b)\quad g(y)\ \text{is}\ p\ \text{times differentiable with respect to}\ y,\\
&\text{and}\ \lim_{y\to 0^+}\frac{\partial^k g(y)}{\partial y^k}=0,\ k=1,2,...,p-1.
\end{aligned} \tag{7}$$

Then the function $h(x,a,b)$ is a p times differentiable jump function.

Proof. See Appendix A.

Remark 1. Several examples of the function $g(y)$ are $g(y)=y^p$, $g(y)=\tanh(y)^p$, $g(y)=\arctan(y^p)$ for any positive integer p, and $g(y)=\sin(y)^p$ for any even positive integer p.

Corollary 1. If the function $g(y)$ in (7) is taken as $g(y)=\exp(-\frac{1}{y})$ then the function $h(x,a,b)$ defined in (5) is a smooth jump function.

Proof. See Appendix B.

4 Control design

To achieve the control objective, we design the control u_i for the robot i based on the new potential function φ. This potential function must attain its unique minimum value when all robots are at their desired positions, and must equal infinity when there is a collision between any robots. The potential function φ should also be chosen such that the gradient based control u_i for the robot i can handle the limited sensing range of the robot. As such, we propose the following potential function

$$\varphi=\frac{e^{\gamma}}{\beta^{\kappa}}-1 \tag{8}$$

where κ is a positive constant, γ and β are the goal and collision avoidance functions. These functions are specified as follows:
-The goal function is designed such that it puts penalty on stabilization errors for all robots, and is equal to zero when the robots are at their final positions. A simple choice of this function is

$$\gamma = \frac{1}{2}\sum_{i=1}^{N} \|q_i - q_{if}\|^2. \tag{9}$$

-The collision function β is chosen such that it equals zero when there is a collision between any robots, and equals 1 when the robots are at their desired positions. We choose this function as follows:

$$\beta = \prod_{i,j} \beta_{ij}, i = 1...N-1, j = i+1,...,N. \tag{10}$$

The function $\beta_{ij} = \beta_{ji}$ is designed as

$$\beta_{ij} = h(0, b_{ij}^2/2, \|q_{ij}\|^2/2) \tag{11}$$

where $h(0, b_{ij}^2/2, \|q_{ij}\|^2/2)$ is a smooth or $p > 2$ times differentiable jump function presented in the previous section, $q_{ij} = q_i - q_j$, b_{ij} is a strictly positive constant such that $b_{ij} \leq \min(R_i, R_j, \varepsilon_2)$ with ε_2 given in (2).

Remark 2. Thanks to properties of the smooth or $p > 2$ differentiable jump function (see Definition 1), the collision function β equals zero when a collision between any robots occurs, i.e. $\|q_{ij}\| = 0$ for any $i \neq j$. The function β equals 1 when all robots are at their desired locations, i.e. $q_i = q_{if}$ for $i = 1,...,N$. The function β is at least twice differentiable with respect to q_{ij}. Hence, the choice of the goal function γ in (9) and the collision function β in (10) with its components given in (11) ensures that the potential function φ in (8) attains the (unique) minimum value of zero when all the robots are at their desired positions, and equals infinity whenever a collision between any robots occurs. Moreover, the potential function φ is at least twice differentiable.

The derivative of φ along the solutions of (1) satisfies

$$\begin{aligned}\dot{\varphi} &= \frac{e^{\gamma}\dot{\gamma}\beta^{\kappa} - e^{\gamma}\kappa\beta^{\kappa-1}\dot{\beta}}{\beta^{2\kappa}} = \frac{e^{\gamma}}{\beta^{\kappa}}\left(\dot{\gamma} - \kappa\sum_{i=1}^{N-1}\sum_{j=i+1}^{N}\frac{\dot{\beta}_{ij}}{\beta_{ij}}\right) \\ &= \frac{e^{\gamma}}{\beta^{\kappa}}\sum_{i=1}^{N}\Omega_i^T u_i \end{aligned} \tag{12}$$

where we have used $\dot{\beta} = \beta\sum_{i=1}^{N-1}\sum_{j=i+1}^{N}\frac{\dot{\beta}_{ij}}{\beta_{ij}}$, and

$$\Omega_i = q_i - q_{if} - \kappa\sum_{j\in\mathbb{N}_i}\bar{\beta}_{ij}' q_{ij} \tag{13}$$

with $\bar{\beta}_{ij}' = \frac{\beta_{ij}'}{\beta_{ij}}$, $\beta_{ij}' = \frac{\partial\beta_{ij}}{\partial(\|q_{ij}\|^2/2)}$, and $\mathbb{N}_i$ the set of all robots, denoted by $\mathbb{N}$, in the group except for the robot i. From (12), a bounded control u_i for the robot i is simply designed as follows:

$$u_i = -c\Psi(\Omega_i) \tag{14}$$

where c is a positive constant, and $\Psi(\Omega_i)$ denotes a vector of bounded functions of elements of Ω_i in the sense that $\Psi(\Omega_i) = \left[\psi(\Omega_i^1)\ \psi(\Omega_i^2),...,\psi(\Omega_i^l),\ ....,\psi(\Omega_i^n)\right]^T$ with Ω_i^l the l^{th} element of Ω_i, i.e.

$\Omega_i = [\Omega_i^1\ \Omega_i^2 ... \Omega_i^l ... \Omega_i^n]^T$. The function $\psi(x)$ is a scalar, differentiable and bounded function, and satisfies

$$\begin{array}{ll} 1) & |\psi(x)| \leq M_1, \\ 2) & \psi(x) = 0 \quad \text{if } x = 0, \quad x\psi(x) > 0 \text{ if } x \neq 0, \\ 3) & \psi(-x) = -\psi(x), (x-y)[\psi(x) - \psi(y)] \geq 0, \\ 4) & |\frac{\psi(x)}{x}| \leq M_2, |\frac{\partial \psi(x)}{\partial x}| \leq M_3, \frac{\partial \psi(x)}{\partial x}|_{x=0} = 1 \end{array} \tag{15}$$

for all $x \in \mathbb{R}, y \in \mathbb{R}$, where M_1, M_2, M_3 are strictly positive constants. Some functions that satisfy the above properties are $\arctan(x)$ and $\tanh(x)$. Indeed, the control u_i is bounded, i.e. $\|u_i(t)\| \leq c\sqrt{n}M_1 := \delta, \forall t \geq t_0 \geq 0$.

Remark 3. When Ω_i defined in (13) is substituted into (14), the l^{th} element of the control u_i can be written as $u_i^l = c\psi\big(-(q_i^l - q_{if}^l) - \sum_{j\in\mathbb{N}_i} \bar{\beta}_{ij}' q_{ij}^l\big)$ with q_i^l, q_{if}^l and q_{ij}^l being the l^{th} elements of q_i, q_{if}, and q_{ij}. The argument of ψ consists of two parts: $-(q_i^l - q_{if}^l)$ and $-\sum_{j\in\mathbb{N}_i} \bar{\beta}_{ij}' q_{ij}^l$. The first part, $-(q_i^l - q_{if}^l)$, referred to as the attractive force plays the role of forcing the robot to its desired location. The second part, $-\sum_{j\in\mathbb{N}_i} \bar{\beta}_{ij}' q_{ij}^l$, referred to as the repulsive force, takes care of collision avoidance for the robot i with the other robots. Moreover, the control u_i of the robot i given in (14) depends on only its own state, and the states of other neighbor robots j if these robots are in a sphere, which is centered at the robot and has a radius no greater than R_i because outside this sphere $\bar{\beta}_{ij}' = 0$.

Now substituting (14) into (12) results in

$$\dot{\varphi} = -c\frac{e^\gamma}{\beta^\kappa} \sum_{i=1}^{N} \Omega_i^T \Psi(\Omega_i). \tag{16}$$

Substituting (14) into (1) results in the closed loop system

$$\dot{q}_i = -c\Psi(\Omega_i), i = 1, ..., N. \tag{17}$$

Theorem 1. Assume that at the initial time $t_0 \geq 0$ each robot starts at a different location, and that each robot has a different desired location, i.e. the conditions given in (2) hold, the bounded controls given in (14) guarantee that no collisions between any robots can occur, the solutions of the closed loop system (17) exist and the robots asymptotically approach their desired positions (a set of equilibria) defined by $q_{if}, i = 1, ..., N$.

Proof. See Appendix C.

5 Simulations

We carry out a simulation with $n = 2, N = 10$. The robots are initialized randomly in a circle, which is centered at the origin and has a radius of 1. The desired formation is specified in shape, location and orientation as $q_{if} = R_f[\sin((i-1)2\pi/N); \cos((i-1)2\pi/N)]$, $i = 1, ..., N$ with $R_f = 8$, i.e. the desired formation is a polygon whose vertices are uniformly distributed on a circle, which is centered at the origin and has a radius of R_f. All robots have the same sensing range: $R_i = 2$. The parameters of the p times differential jump functions are $p = 2, b_{ij} = 1, g(y) = y^p$. The function ψ is taken as arctan. The control gains are chosen as $\kappa = 1, c = 2$. Simulation results are plotted in Fig. 1. It is seen that all robots nicely approach their desired locations. Since the robots initialize pretty close to each other, they quickly move away from each other then approach their desired locations, see Sub-figure A) of Fig. 1, where the trajectory of the robot 1 is plotted in the thick line, and robots 1 and 2 are indicated by 1 and 2. For clarity, only the control input $u_1 = [u_{1x}\ u_{1y}]^T$ is plotted in Sub-figure B) of Fig. 1. Noticing that

the values of the continuous controls u_{1x} and u_{1y} are in the range $\pm\pi$. Sub-figure C) of Fig.1 plot the functions $\beta_{1j}, j = 2,...,N$. It is seen that these functions are always greater than zero and approach 1. Sub-figure D) of Fig. 1 plots a 'mean-product' distance, $\text{dist}_{\text{all}} = \left(\prod_{(i,j)\in\mathbb{N}} \|q_{ij}\|\right)^{N(N-1)/2}$, see the thick line, and the distances between the robot 1 and other robots in the group. Clearly, no collisions between any robots occurred since dist_{all} is larger than zero for all simulation time. The bottom figure in Fig. 1 plots the functions $\beta_{1j}, j = 2,...,N$. It is noted that all β_{1j} equal 1 when $\|q_{1j}\|$ are larger or equal to 1.

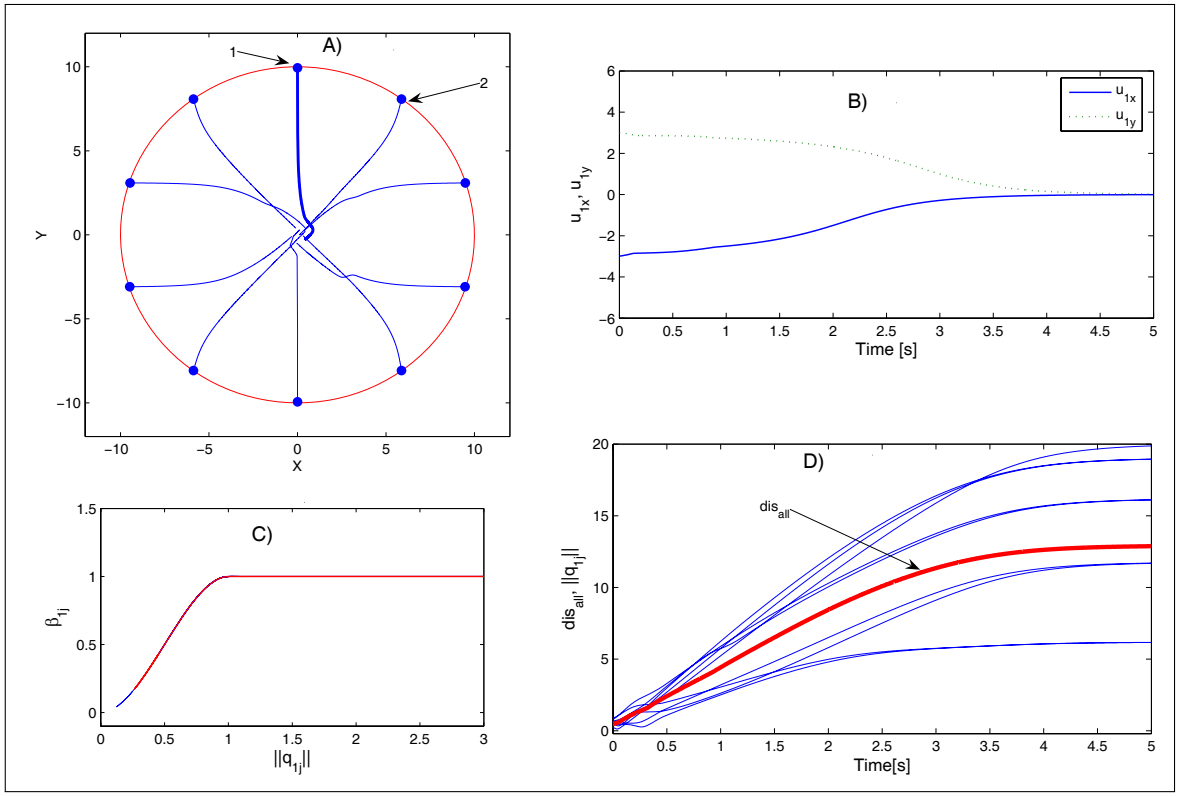


Figure 1: Simulation results.

6 Extension to formation tracking

This section extends the results developed in the previous sections to solve the problem of designing a control input u_i for each robot i that forces the group of N mobile robots whose dynamics are given in (1) to track a moving Desired Formation Graph (DFG), in the sense that the DFG is allowed to move on a common desired trajectory $q_{od}(s)$ with s being the common reference trajectory parameter defined in the fixed coordinate system Π_F, see Fig. 2 and Fig. 3. We consider the DFG whose center moves along the common reference trajectory $q_{od}(s)$. We assume that $q_{od}(s)$ is regular in the sense that it is single valued and its first derivative exists and is bounded. Since the DFG under consideration is only representative, the center does not have to be the "true" center of the DFG but can be any convenient point. When the DFG moves along the trajectory $q_{od}(s)$, the vertex i of DFG generates the reference trajectory $q_{id}(s)$ for the robot i to track. We limit our consideration to two- and three-dimensional (2D and 3D) spaces, which are most common in practice. The control objective is now stated as follows.

Control objective. Assume that at the initial time $t_0 \geq 0$, for all $(i, j) \in \{1,2,...N\}, i \neq j, s \in \mathbb{R}$ there

exist strictly positive constants ε_1, ε_2 and ε_3 such that

$$\|q_i(t_0) - q_j(t_0)\| \geq \varepsilon_1,\ \|q_{id}(s) - q_{jd}(s)\| \geq \varepsilon_2,$$
$$\|\frac{\partial q_{id}(s)}{\partial s}\| \leq \varepsilon_3 \tag{18}$$

Moreover, the robot i can only measure its own state and can only detect the other group members if these members are in a sphere with a radius of R_i larger than a strictly positive constant, and centered at the robot i. Design the control input u_i for each robot i such that

$$\lim_{t\to\infty}(q_i(t) - q_{id}(s)) = 0, \quad \|q_i(t) - q_j(t)\| \geq \varepsilon_4 \tag{19}$$

for all $(i, j) \in \{1, 2, ...N\}, i \neq j, s \in \mathbb{R}$, where ε_4 is a positive constant.

Control design. Let us construct $q_{id}(s)$. Let the moving coordinate system Π_M, of which the origin $\widehat{O}$ coincides with the center of the desired formation graph, move along $q_{od}(s)$. Let $\widehat{q}_{id}$ be the coordinate vector of the vertex i of the DFG in the moving coordinate system. We then have

$$\widehat{q}_{id} = J(\bullet)(q_{id}(s) - q_{od}(s)) \tag{20}$$

where $J(\bullet)$ whose elements depend on $\partial q_{od}(s)/\partial s$ and is the rotational (invertible) matrix, which describes the rotation of Π_M with respect to Π_F, and is such that $\delta_1 \leq \|J(\bullet)\| \leq \delta_2$, $\delta_3 \leq \|J(\bullet)^{-1}\| \leq \delta_4$ with $\delta_i, i = 1, .., 4$ strictly positive constants. Therefore, by specifying $\widehat{q}_{id}$ in Π_M, $q_{id}(s)$ in Π_F for the robot i can be calculated from (20). Similarly, in Π_M the coordinate vector of each robot i satisfies

$$\widehat{q}_i = J(\bullet)(q_i - q_{od}(s)). \tag{21}$$

From (20) and (21), we can see that the first two conditions in (18) imply the following condition

$$\|\widehat{q}_i(t_0) - \widehat{q}_j(t_0)\| \geq \widehat{\varepsilon}_1,\ \|\widehat{q}_{id}(s) - \widehat{q}_{jd}(s)\| \geq \widehat{\varepsilon}_2 \tag{22}$$

where $\widehat{\varepsilon}_1$ and $\widehat{\varepsilon}_2$ are some strictly positive constants. Moreover, the tracking control goal specified in (19) is achieved by designing the control u_i for each robot i such that

$$\lim_{t\to\infty}(\widehat{q}_i(t) - \widehat{q}_{id}(s)) = 0, \quad \|\widehat{q}_i(t) - \widehat{q}_j(t)\| \geq \widehat{\varepsilon}_4 \tag{23}$$

where $\widehat{\varepsilon}_4$ is a positive constant, and by letting the DFG move along the common reference trajectory via giving $\dot{s}$ some desired value.

Now differentiating both sides of (21) gives

$$\dot{\widehat{q}}_i = \widehat{u_i} \tag{24}$$

where $\widehat{u_i}$ is the new control, and we have chosen the control u_i as

$$u_i = \dot{q}_{od}(s) + J(\bullet)^{-1}(\widehat{u_i} - \dot{J}(\bullet)(q_i - q_{od}(s))). \tag{25}$$

The problem of designing $\widehat{u_i}$ for (24) to achieve (23) under (22) is exactly the same as the control objective in Section 2. Therefore, the control design in Section 4 can be used directly to design a bounded control $\widehat{u_i}$ to achieve the goal (23). After $\widehat{u_i}$ is designed, the actual tracking control u_i is calculated from (25). Let us give the expression of the rotational matrix $J(\bullet)$ in 2D and 3D spaces.

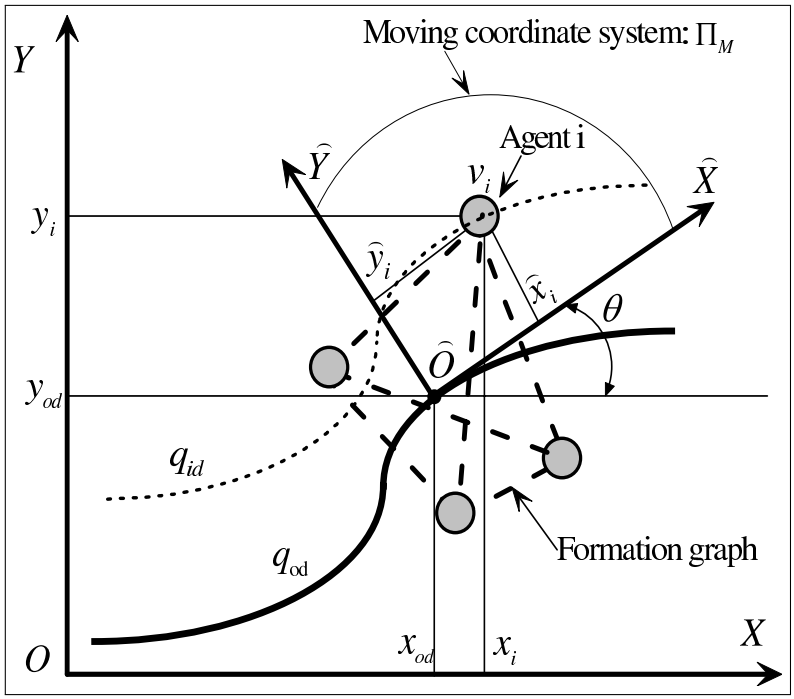


Figure 2: Formation coordinates in 2D.

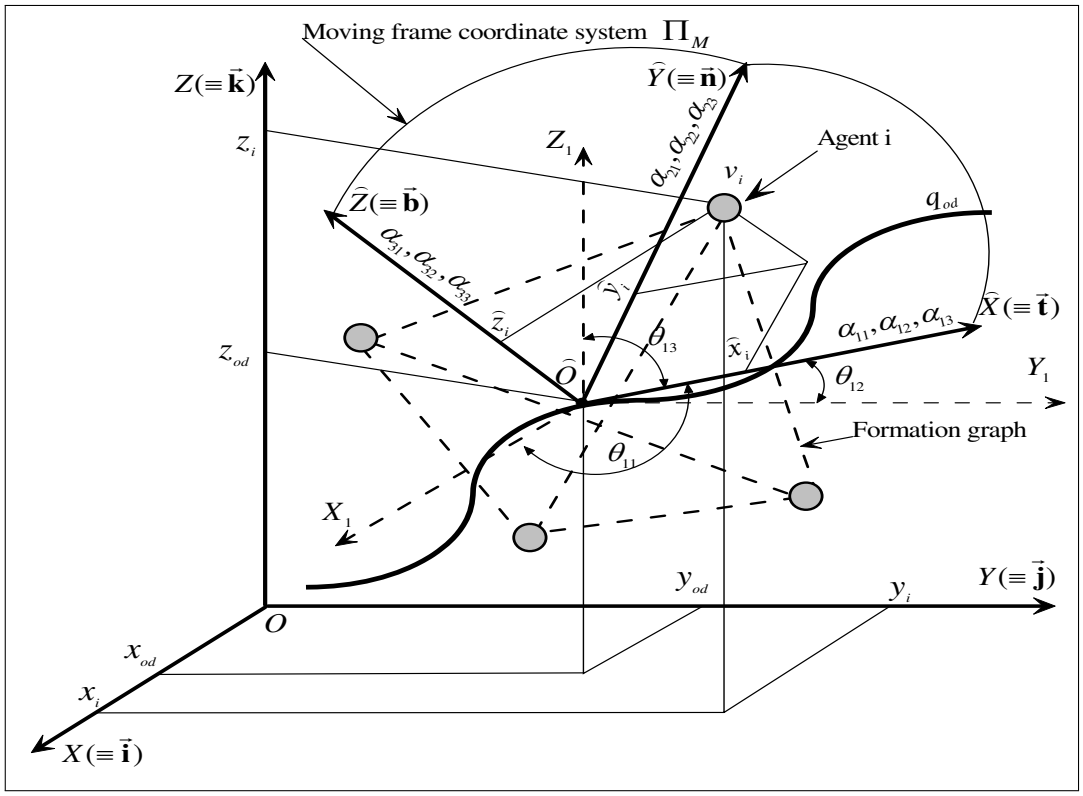


Figure 3: Formation coordinates in 3D.

Two-dimensional space. Consider the moving coordinate frame, $\widehat{O}\widehat{X}\widehat{Y}$ attached to the DFG, as shown in Fig. 2. The origin $\widehat{O}$ coincides with the center of the graph, and is on the common reference trajectory $q_{od}(s) = [x_{od}(s)\ y_{od}(s)]^T$. The $\widehat{O}\widehat{X}$ and $\widehat{O}\widehat{Y}$ axes are tangential and perpendicular to the reference trajectory $q_{od}(s)$. Therefore the angle θ between $\widehat{O}\widehat{X}$ and OX is calculated as $\theta = \arctan(y_d^{'}/x_d^{'})$, where $\bullet' \triangleq \partial \bullet / \partial s$. Hence the rotational matrix $J(\bullet)$ is given by $J(\bullet) = \begin{bmatrix} \cos(\theta) & \sin(\theta) \\ -\sin(\theta) & \cos(\theta) \end{bmatrix}$, which is indeed invertible for all $\theta \in \mathbb{R}$, and $\|J(\bullet)\| = 1$.

Three-dimensional space. Consider the moving coordinate frame, $\widehat{O}\widehat{X}\widehat{Y}\widehat{Z}$, attached to the DFG as shown in Fig. 3. The coordinate frame $\widehat{O}\ X_1Y_1Z_1$ is parallel to $OXYZ$. The origin $\widehat{O}$ coincides with the center of the graph, and is on the reference trajectory $q_{od}(s) = [x_{od}(s)\ y_{od}(s)\ z_{od}(s)]^T$. The $\widehat{O}\widehat{X}$, $\widehat{O}\widehat{Y}$ and $\widehat{O}\widehat{Z}$ axes coincide with the unit tangent vector $\vec{t}$, the unit principal vector $\vec{n}$, and the unit binormal vector $\vec{b}$ of the trajectory $q_{od}(s)$ at the point $\widehat{O}$. These unit vectors form a positively oriented triple of vectors called the moving triad, and are given by $\vec{t} = \vec{q}^{'}_{od}/\|\vec{q}^{'}_{od}\|$, $\vec{n} = \vec{t}^{'}/\|\vec{t}^{'}\|$, $\vec{b} = \vec{t} \times \vec{n}$, where $\times$ stands for the vector cross product operation, $(\vec{i},\ \vec{j},\ \vec{k})$ are the unit vectors of the $OXYZ$ coordinate frame. Let $(\xi_{i1},\ \xi_{i2}, \xi_{i3}), i = 1,2,3$ be the directional cosines of $\widehat{O}\widehat{X}$, $\widehat{O}\widehat{Y}$ and $\widehat{O}\widehat{Z}$ with respect to the fixed axes OX, OY and OZ, respectively. This notation means that if we denote $(\theta_{i1},\ \theta_{i2}, \theta_{i3}), i = 1,2,3$ as the angles between the axes $\widehat{O}\widehat{X}$, $\widehat{O}\widehat{Y}$ and $\widehat{O}\widehat{Z}$, and the axes OX, OY and OZ (see Fig. 3 for representation of θ_{11}, θ_{12} and θ_{13}), we have $\xi_{ij} = \cos(\theta_{ij})$, $\forall i, j \in \{1,2,3\}$. The rotational matrix $J(\bullet)$ is given by $J(\bullet) = \begin{bmatrix} \xi_{11} & \xi_{12} & \xi_{13} \\ \xi_{21} & \xi_{22} & \xi_{23} \\ \xi_{31} & \xi_{32} & \xi_{33} \end{bmatrix}$. It is shown in [20] that the determinant of $J(\bullet)$ is equal to 1, i.e. $J(\bullet)$ is globally invertible, and $\|J(\bullet)\| = 1$.

7 Summary and Conclusions

This paper has presented a method to design bounded and continuous (even smooth) controllers to force a group of mobile robots with limited sensing ranges to achieve a desired formation while avoiding collisions among themselves. The control development was based on construction of new potential functions, and guaranteed that all critical points, except for the desired points in formation, are unstable points. These potential functions with embedded smooth or p times differential jump functions are attractive parts of the paper since they do not require the use of switching control theory despite the robot limited sensing ranges. These functions can certainly be modified to solve other cooperative control problems such as flocking and consensus of mobile robots. The problem of formation tracking was also addressed. Future work is to extend the proposed techniques in this paper and those for centralized formation control of nonholonomic mobile robots in [10] to design a decentralized formation control system for a group of nonholonomic mobile robots.

8 Appendix A: Proof of Lemma 1

We need to verify that the function $h(x,a,b)$ given in (5) satisfy all properties defined in (4). Property 1) holds because by (6) for all $0 \le x \le a$, we have $\int_a^x f(\tau - a) f(b - \tau) d\tau = 0$. Property 2) holds since by (6) we have $\int_a^x f(\tau - a) f(b - \tau) d\tau = \int_a^b f(\tau - a) f(b - \tau) d\tau$ for all $x \ge b$. To prove Property 3), we first note from Property a) of the function $g(y)$ given in (7) that $\int_a^x f(\tau - a) f(b - \tau) d\tau > 0$ for all $a < x < b$. Therefore, $0 < \frac{\int_a^x f(\tau-a)f(b-\tau)d\tau}{\int_a^b f(\tau-a)f(b-\tau)d\tau} < 1$, which means that Property 3) of the function $h(x,a,b)$ holds. To prove Property 4), we just need to show that $f(y)$ is $p-1$ times differentiable. We first note that $f(y)$ is

p times differentiable except at $y = 0$. Hence, we only need to verify that $f^{(k)}(0) = \frac{\partial^k f(y)}{\partial y^k}\Big|_{y=0} = 0$ for any positive integer $k < p$. Clearly, $\lim_{y\to 0^-} f^{(k)}(y) = 0$ since $f(y) = 0$, $\forall y \leq 0$. On the other hand, since $f(y) = g(y), y > 0$, from Property b) of the function $g(y)$ we have $\lim_{y\to 0^+} f^{(k)}(y) = \lim_{y\to 0^+} g^{(k)}(y) = 0$, where $g^{(k)} = \frac{\partial^k g(y)}{\partial y^k}$. Since both left- and right-hand limits are equal to 0, we have $f^{(k)}(0) = 0$. Hence Property 4) holds. □

9 Appendix B: Proof of Corollary 1

We first note that Property a) of the function $g(y)$ in (7) can be proven without a difficulty. We focus on proof of Property b). We note that $g^{(k)}(y) = \frac{\partial^k g(y)}{\partial y^k} = Q_k(\frac{1}{y})e^{-\frac{1}{y}}$ where $Q_k(\frac{1}{y})$ is a polynomial function of $\frac{1}{y}$, and k is any positive integer. We will prove Property b) of the function $g(y)$ in (7) by induction. It is clear that $\lim_{y\to 0^+} g^{(1)}(y) = \lim_{y\to 0^+} \frac{g(y)-g(0)}{y-0} = \lim_{y\to 0^+} \frac{e^{-\frac{1}{y}}}{y} = \lim_{\xi\to\infty} \frac{1}{e^{\xi}} = 0$ where $\xi = \frac{1}{y}$ and we have used l'Hopital's rule. This means that Property b) of the function $g(y)$ holds for $k = 1$. Assuming that $\lim_{y\to 0^+} g^{(k)}(y) = 0$, we now compute $\lim_{y\to 0^+} g^{(k+1)}(y)$ as follows:

$$\lim_{y\to 0^+} g^{(k+1)}(y) = \lim_{y\to 0^+} \frac{g^{(k)}(y) - g^{(k)}(0)}{y-0} =$$
$$\lim_{y\to 0^+} \tilde{Q}_k(\frac{1}{y})e^{-\frac{1}{y}} = \lim_{\xi\to\infty} \frac{\tilde{Q}_k(\xi)}{e^{\xi}} = 0 \tag{26}$$

where l'Hopital's rule has been used, $\xi = \frac{1}{y}$, and $\tilde{Q}_k(\xi) = \xi Q_k(\xi)$ is another polynomial of ξ. Therefore we have proved that $\lim_{y\to 0^+} g^{(k)}(y) = 0$ for any k, which means Property b) of the function $g(y)$ holds for any positive integer p, i.e. p can be equal to infinity. By Definition 1, the function $h(x,a,b)$ is a smooth jump function.□

10 Appendix C: Proof of Theorem 1

We prove Theorem 1 in two steps. In the first step, we prove that no collisions between the robots can occur and that the robots asymptotically approach their target points or some critical points. In the second step, we investigate stability of the closed loop system (17) at these points, we linearize the closed loop system at these points. We then prove that only desired target points are unique asymptotic stable and that other critical points are unstable.

Step 1. Proof of no collision and existence of solutions. From (16) and properties of the function ψ , see (15), we have $\dot{\varphi} \leq 0$, which implies that $\varphi(t) \leq \varphi(t_0), \forall t \geq t_0$. With definition of the function φ in (8) and its components in (9) and (10), we have

$$\frac{e^{0.5\sum_{i=1}^{N} \|q_i(t)-q_{if}\|^2}}{\prod h(0, b_{ij}^2/2, \|q_{ij}(t)\|^2/2)} \leq \frac{e^{0.5\sum_{i=1}^{N} \|q_i(t_0)-q_{if}\|^2}}{\prod h(0, b_{ij}^2/2, \|q_{ij}(t_0)\|^2/2)} \tag{27}$$

for all $t \geq t_0 \geq 0, i = 1,...,N-1, j = i+1,...N$. From the first condition in (2) and properties of the jump function $h(0, b_{ij}^2/2, \|q_{ij}\|^2/2)$, we have $\prod h(0, b_{ij}^2/2, \|q_{ij}(t_0)\|^2/2)$ is larger than a strictly positive constant. Therefore the right hand side of (27) is bounded by a positive constant depending on the initial conditions. Boundedness of the right hand side of (27) implies that the left hand side of (27) must be also bounded. As a result, $\prod h(0, b_{ij}^2/2, \|q_{ij}(t)\|^2/2)$ must be larger than some positive constant depending

on the initial conditions for all $t \geq t_0 \geq 0$. From properties of $h(0, b_{ij}^2/2, \|q_{ij}\|^2/2)$, $\|q_{ij}(t)\|$ must be larger than some positive constant depending on the initial conditions denoted by ε_3, i.e. there are no collisions for all $t \geq t_0 \geq 0, i = 1,...,N-1, j = i+1,...N$. Boundedness of the left hand side of (27) also implies that of $\|q_{ij}(t)\|$ and $\|q_i(t)\|$ for all $t \geq t_0 \geq 0$, i.e. the solutions of the closed loop system (17) exist.

+*Equilibrium points*. Since we have already proved that there are no collisions between any robots and that the solutions of the closed loop system (17) exist, an application of Theorem 8.4 in [19] to (16) yields

$$\lim_{t\to\infty} \Omega_i^T(t)\Psi(\Omega_i(t)) = 0, \forall i = 1,2,...,N. \tag{28}$$

Thanks to Property 2) of the function ψ, see (15), the limit equation (28) implies that

$$\lim_{t\to\infty} \Omega_i(t) = \lim_{t\to\infty} \Big[q_i(t) - q_{if} - \kappa \sum_{j\in\mathbb{N}_i} \bar{\beta}'_{ij}(t) q_{ij}(t) \Big] = 0 \tag{29}$$

for all $i = 1,2,...,N$. The limit equation (29) implies that the state $q(t) = [q_1^T(t)\, q_2^T(t), ..., q_N^T(t)]^T$ converges to the manifold $\mathbb{M}$ of (17) contained in $\mathbb{E} = \{q \in \mathbb{R}^{n\times N}|_{\Omega=0}\}$ with $\Omega = [\Omega_1^T\, \Omega_2^T, ..., \Omega_N^T]^T$, i.e. on the surface where $\dot{\varphi} = 0$. This surface is continuous because we have already proved that $\|q_{ij}\| > 0, \forall (i,j) \in \{1,2,...,N\}, i \neq j$, i.e. $\bar{\beta}'_{ij}$ is continuous. As the time t goes to infinity, it can be verified that one solution of (29) is $q_f = [q_{1f}^T\, q_{2f}^T, ..., q_{Nf}^T]^T$ since $\beta'_{ij}|_{\|q_{ij}\|=\|q_{ijf}\|} = 0 \Rightarrow \bar{\beta}'_{ij}|_{\|q_{ij}\|=\|q_{ijf}\|} = 0$ (we have chosen $b_{ij} \leq \min(R_i, R_j, \varepsilon_2)$, see (11)), and other solutions are denoted by $q_c = [q_{1c}^T\, q_{2c}^T, ..., q_{Nc}^T]^T$. It is noted that some elements of q_c can be equal to that of q_f. However, for simplicity we abuse the notation, i.e. we still denote that vector as q_c. Indeed, the vector q_c is such that

$$\Omega_i|_{q=q_c} = \big[q_i - q_{if} - \kappa \sum_{j\in\mathbb{N}_i} \bar{\beta}'_{ij} q_{ij} \big]\Big|_{q=q_c} = 0 \tag{30}$$

for all $i = 1,...,N$. Next, we will show that q_f is stable and q_c is unstable, by linearizing (17) at these points.

+*Properties of equilibrium points*. The closed loop system (17) can be written in a vector form as $\dot{q} = -c\Psi_q(q, q_f)$, and $\Psi_q(q, q_f) = [\Psi^T(\Omega_1),\ \ ..., \Psi^T(\Omega_i), ..., \Psi^T(\Omega_N)]^T$. Therefore, near an equilibrium point q_o, which can be either q_f or q_c, we have

$$\dot{q} = -c\, \partial\Psi_q(q, q_f)/\partial q\big|_{q=q_o} (q - q_o) \tag{31}$$

where

$$\frac{\partial \Psi_q(q, q_f)}{\partial q} = \begin{bmatrix} \Delta_{11} & \Delta_{12} & \cdots & \cdots & \Delta_{1N} \\ \vdots & \ddots & \vdots & \vdots & \vdots \\ \Delta_{i1} & \cdots & \Delta_{ii} & \cdots & \Delta_{iN} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \Delta_{N1} & \cdots & \cdots & \cdots & \Delta_{NN} \end{bmatrix} \tag{32}$$

with $\Delta_{ij} = \frac{\partial \Psi(\Omega_i)}{\partial \Omega_i} \frac{\partial \Omega_i}{\partial q_i}, (i,j) \in \mathbb{N}$, where $\mathbb{N}$ denotes the set of all robots in the group. A simple calculation shows that for all $i = 1,...,N, j \in \mathbb{N}_i, j \neq i$

$$\begin{aligned} \frac{\partial \Omega_i}{\partial q_i} &= \Big(1 - \kappa \sum_{i\in\mathbb{N}_i} \bar{\beta}'_{ij}\Big) I_n - \kappa \sum_{j\in\mathbb{N}_i} \bar{\beta}''_{ij} q_{ij} q_{ij}^T, \\ \frac{\partial \Omega_i}{\partial q_j} &= \kappa \bar{\beta}'_{ij} I_{n\times n} + \kappa \bar{\beta}''_{ij} q_{ij} q_{ij}^T \end{aligned} \tag{33}$$

where $\bar{\beta}''_{ij} = \frac{\partial \bar{\beta}'_{ij}}{\partial(\|q_{ij}\|^2/2)}$. Let $\mathbb{N}^*$ be the set of the robots such that if the robots i and j belong to the set $\mathbb{N}^*$ then $\|q_{ij}\| < b_{ij}$. Next we will show that q_f is asymptotically stable and that q_c is unstable.

Step 2. Behavior of equilibrium points. Evaluating (33) at $q = q_f$ gives

$$\left.\frac{\partial \Psi(\Omega_i)}{\partial \Omega_i}\right|_{q=q_f} = I_n, \left.\frac{\partial \Omega_i}{\partial q_i}\right|_{q=q_f} = I_n, \left.\frac{\partial \Omega_i}{\partial q_j}\right|_{q=q_f} = 0 \tag{34}$$

where we have used $\bar{\beta}'_{ij}\big|_{q_{ij}=q_{ijf}} = 0$ and $\bar{\beta}''_{ij}\big|_{q_{ij}=q_{ijf}} = 0$ since $\beta'_{ij}\big|_{q_{ij}=q_{ijf}} = 0$ and $\beta''_{ij}\big|_{q_{ij}=q_{ijf}} = 0$ (we have chosen $b_{ij} \leq \min(R_i, R_j, \varepsilon_2)$, see (11)). We consider the Lyapunov function candidate $V_f = 0.5\|q - q_f\|^2$ whose derivative along the solutions of the linearized closed loop system (31) with q_o replaced by q_f, and using (34) satisfies $\dot{V}_f = -c\sum_{i=1}^{N} \|q_i - q_{if}\|^2 = -2cV_f$, which implies that q_f is asymptotically stable.

- Proof of q_c being unstable. Now evaluating (33) at $q = q_c$ give that for all $i = 1,...,N, i \neq j$:

$$\begin{aligned}
&\left.\frac{\partial \Psi(\Omega_i)}{\partial \Omega_i}\right|_{q=q_c} = I_n, \left.\frac{\partial \Omega_i}{\partial q_i}\right|_{q=q_c} = \big(1 - \kappa \sum_{j\in\mathbb{N}_i} \bar{\beta}'_{ijc}\big) I_n - \\
&\kappa \sum_{j\in\mathbb{N}_i} \bar{\beta}''_{ijc} q_{ijc} q_{ijc}^T, \left.\frac{\partial \Omega_i}{\partial q_j}\right|_{q=q_c} = \kappa \bar{\beta}'_{ijc} + \kappa \bar{\beta}''_{ijc} q_{ijc} q_{ijc}^T
\end{aligned} \tag{35}$$

where $q_{ijc} = q_{ic} - q_{jc}$, $\bar{\beta}'_{ijc} = \bar{\beta}'_{ij}\big|_{q_{ij}=q_{ijc}}$ and $\bar{\beta}''_{ijc} = \bar{\beta}''_{ij}\big|_{q_{ij}=q_{ijc}}$. Since the related collision avoidance functions β_{ij}, (hence $\bar{\beta}'_{ij}$ and $\bar{\beta}''_{ij}$), are specified in terms of relative distances between robots and it is extremely difficult to obtain q_c explicitly by solving (30), it is very difficult to use the Lyapunov function candidate $V_c = 0.5\|q - q_c\|$ to investigate stability of (31) at q_c. Therefore, we consider the following Lyapunov function candidate

$$\bar{V}_c = 0.5\|\bar{q} - \bar{q}_c\|^2 \tag{36}$$

where $\bar{q} = [q_{12}^T, q_{13}^T, ...q_{1N}^T, q_{23}^T, ..., q_{2N}^T, ..., q_{N-1,N}^T]^T$ and $\bar{q}_c = [q_{12c}^T, q_{13c}^T, ...q_{1Nc}^T, q_{23c}^T, ..., q_{2Nc}^T, ..., q_{N-1,Nc}^T]^T$. Differentiating both sides of (36) along the solution of (31) with q_o replaced by q_c gives

$$\begin{aligned}
\dot{\bar{V}}_c = &-c \sum_{(i,j)\in\mathbb{N}\setminus\mathbb{N}^*} \|q_{ij} - q_{ijc}\|^2 - c \sum_{(i,j)\in\mathbb{N}^*} (1 - \kappa N \bar{\beta}'_{ijc}) \times \\
&\|q_{ij} - q_{ijc}\|^2 + \kappa c N \sum_{(i,j)\in\mathbb{N}^*} \bar{\beta}''_{ijc} \big(q_{ijc}^T (q_{ij} - q_{ijc})\big)^2
\end{aligned} \tag{37}$$

where $i \neq j$ and (35) has been used. To investigate stability properties of $\bar{q}_c$ based on (37), we will use (30). Define $\Omega_{ijc} = \Omega_{ic} - \Omega_{jc}$, $\forall (i,j) \in \{1,...,N\}$, $i \neq j$ where $\Omega_{ic} = \Omega_i|_{q=q_c} = 0$, see (30). Therefore $\Omega_{ijc} = 0$. Hence $\sum_{(i,j)\in\mathbb{N}^*} q_{ijc}^T \Omega_{ijc} = 0, i \neq j$, which by using (30) is expanded to

$$\begin{aligned}
&\sum_{(i,j)\in\mathbb{N}^*} \big(q_{ijc}^T (q_{ijc} - q_{ijf}) - \kappa N \bar{\beta}'_{ijc} q_{ijc}^T q_{ijc}\big) = 0 \\
&\Rightarrow \sum_{(i,j)\in\mathbb{N}^*} (1 - \kappa N \bar{\beta}'_{ijc}) q_{ijc}^T q_{ijc} = \sum_{(i,j)\in\mathbb{N}^*} q_{ijc}^T q_{ijf}
\end{aligned} \tag{38}$$

where $i \neq j$. The sum $\sum_{(i,j)\in\mathbb{N}^*} q_{ijc}^T q_{ijf}$ is strictly negative since at the point where $q_{ij} = q_{ijf}$, $\forall (i,j) \in \mathbb{N}^*, i \neq j$ (the point F in Fig. 4) all attractive and repulsive forces are equal to zero while at the point where $q_{ij} = q_{ijc}$ $\forall (i,j) \in \mathbb{N}^*, i \neq j$ (the point C in Fig. 4) the sum of attractive and repulsive forces are equal to zero (but attractive and repulsive forces are nonzero). Therefore the point where $q_{ij} = 0$, $\forall (i,j) \in \mathbb{N}^*, i \neq j$ (the point O in Fig. 4) must locate between the points F and C for all $(i,j) \in \mathbb{N}^*, i \neq j$. That is

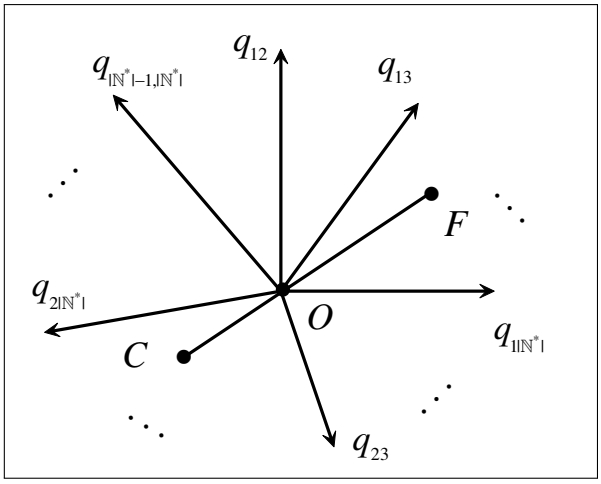


Figure 4: Illustration of equilibrium points.

there exists a strictly positive constant b such that $\sum_{(i,j)\in\mathbb{N}^*} q_{ijc}^T q_{ijf} < -b$, which is substituted into (38) to yield

$$\sum_{(i,j)\in\mathbb{N}^*} (1-\kappa N\bar{\beta}'_{ijc}) q_{ijc}^T q_{ijc} < -b, i \neq j. \tag{39}$$

Since $q_{ijc}^T q_{ijc} > 0, \forall (i,j) \in \mathbb{N}^*, i \neq j$, there exists a nonempty set $\mathbb{N}^{**} \subset \mathbb{N}^*$ such that for all $(i,j) \in \mathbb{N}^{**}, i \neq j$, $(1-\kappa N\bar{\beta}'_{ijc})$ is strictly negative, i.e. there exists a strictly positive constant b^{**} such that $(1-\kappa N\bar{\beta}'_{ijc}) < -b^{**}$, $\forall (i,j) \in \mathbb{N}^{**}, i \neq j$. We now write (37) as

$$\begin{aligned}\dot{\bar{V}}_c = -c\Bigg[&\sum_{(i,j)\in\mathbb{N}\setminus\mathbb{N}^*} \|q_{ij}-q_{ijc}\|^2 + \sum_{(i,j)\in\mathbb{N}^*\setminus\mathbb{N}^{**}} (1-\kappa N\bar{\beta}'_{ijc}) \times \\ &\|q_{ij}-q_{ijc}\|^2 - \kappa N \sum_{(i,j)\in\mathbb{N}^*} \bar{\beta}''_{ijc}\big(q_{ijc}^T(q_{ij}-q_{ijc})\big)^2\Bigg] - \\ &c \sum_{(i,j)\in\mathbb{N}^{**}} (1-\kappa N\bar{\beta}'_{ijc}) \|q_{ij}-q_{ijc}\|^2 \end{aligned} \tag{40}$$

where $i \neq j$. We now define a subspace such that $q_{ij}-q_{ijc}=0$, $\forall (i,j) \in \mathbb{N}\setminus\mathbb{N}^{**}$ and $q_{ijc}^T(q_{ij}-q_{ijc}) = 0$, $\forall (i,j) \in \mathbb{N}^*, i \neq j$. In this subspace, we have

$$\bar{V}_c = 0.5 \sum_{(i,j)\in\mathbb{N}^{**}} \|q_{ij}-q_{ijc}\|^2, \tag{41}$$

$$\dot{\bar{V}}_c = -c \sum_{(i,j)\in\mathbb{N}^{**}} (1-\kappa N\bar{\beta}'_{ijc}) \|q_{ij}-q_{ijc}\|^2 \geq 2cb^{**}\bar{V}_c$$

where we have used $(1-\kappa N\bar{\beta}'_{ijc}) < -b^{**}$, $\forall (i,j) \in \mathbb{N}^{**}, i \neq j$. Clearly (41) implies that

$$\sum_{(i,j)\in\mathbb{N}^{**}} \|q_{ij}(t)-q_{ijc}\| \geq \sum_{(i,j)\in\mathbb{N}^{**}} \|q_{ij}(t_0)-q_{ijc}\| e^{cb^{**}(t-t_0)} \tag{42}$$

for all $i \neq j$, $t \geq t_0 \geq 0$. Now assume that q_c is a stable equilibrium point of the closed loop system (17), i.e. $\lim_{t\to\infty} \|q_i(t)-q_{ic}\| = d_i, \forall i \in \mathbb{N}$ with d_i a nonnegative constant. Note that $\mathbb{N}^{**} \subset \mathbb{N}$, we have $\lim_{t\to\infty} \|q_i(t)-q_{ic}\| = d_i, \forall i \in \mathbb{N}^{**}$, which implies that $\lim_{t\to\infty} \sum_{(i,j)\in\mathbb{N}^{**}} \|q_{ij}(t)-q_{ijc}\| = d^{**}, \forall (i,j) \in \mathbb{N}^{**}, i \neq j$ with d^{**} a nonnegative constant, since $q_{ij} = q_i - q_j$ and $q_{ijc} = q_{ic} - q_{jc}$. This contradicts (42) for the case $\sum_{(i,j)\in\mathbb{N}^{**}} \|q_{ij}(t_0)-q_{ijc}\| \neq 0$, since the right hand side of (42) is divergent (so does the left

hand side). For the case $\sum_{(i,j)\in\mathbb{N}^{**}} \|q_{ij}(t_0) - q_{ijc}\| = 0$, there would be no contradiction. However this case is never observed in practice since the ever-present physical noise would cause $\|q_{ij}(t^*) - q_{ijc}\|$ for some $(i,j) \in \mathbb{N}^{**}, i \neq j$ to be different from 0 at the time $t^* \geq t_0$. We now write (42) as

$$\sum_{(i,j)\in\mathbb{N}^{**}} \|q_{ij}(t) - q_{ijc}\| \geq \sum_{(i,j)\in\mathbb{N}^{**}} \|q_{ij}(t^*) - q_{ijc}\| e^{cb^{**}(t-t^*)} \tag{43}$$

for all $i \neq j$, $t \geq t^* \geq t_0 \geq 0$. Since $\sum_{(i,j)\in\mathbb{N}^{**}} \|q_{ij}(t^*) - q_{ijc}\| \neq 0$, the right hand side of (43) is divergent (so does the left hand side). This contradicts $\lim_{t\to\infty} \sum_{(i,j)\in\mathbb{N}^{**}} \|q_{ij}(t) - q_{ijc}\| = d^{**}, \forall (i,j) \in \mathbb{N}^{**}, i \neq j$. Therefore q_c must be an unstable equilibrium point of the closed loop system (17). Proof of Theorem 1 is completed.

Bibliography

[1] P. K. C. Wang, Navigation Strategies for Multiple Autonomous Mobile Robots Moving in Formation, *J. Robot. Syst.*, Vol. 8, No. 2, pp. 177-195, 1991.

[2] A. K. Das, R. Fierro, V. Kumar, J. P. Ostrowski J. Spletzer, and C. J. Taylor, A Vision Based Formation Control Framework, *IEEE Transactions on Robotics and Automation*, vol. 18, pp. 813-825, 2002.

[3] N. E. Leonard and E. Fiorelli, Virtual Leaders, Artificial Potentials and Coordinated Control of Groups, *Proceedings of IEEE Conference on Decision and Control*, Orlando, FL, pp. 2968-2973, 2001.

[4] R.T. Jonathan, R.W. Beard and B.J. Young, A Decentralized Approach to Formation Maneuvers, *IEEE Transactions on Robotics and Automation*, vol. 19, pp. 933-941, 2003.

[5] T. Balch and R. C. Arkin, Behavior-Based Formation Control for Multirobot Teams, *IEEE Transactions on Robotics and Automation*, vol. 14, pp. 926-939, 1998.

[6] M. A. Lewis and K.-H. Tan, High Precision Formation Control of Mobile Robots Using Virtual Structures, *Autonomous Robots*, vol. 4, pp. 387-403, 1997.

[7] R. Skjetne, Moi, S., and T. I. Fossen, Nonlinear Formation Control of Marine Craft, *Proceedings of IEEE Conference on Decision and Control*, Las Vegas, NV, pp. 1699-1704, 2002.

[8] D. M. Stipanovica, G. Inalhana, R. Teo and C. J. Tomlina, Decentralized Overlapping Control of a Formation of Unmanned Aerial Vehicles, *Automatica*, vol. 40, pp. 1285 -1296, 2004.

[9] D.B. Nguyen and K.D. Do, Formation control of mobile robots, *International Journal of Computers, Communications and Control*, Vol. I, No. 3, pp. 41-59, 2006.

[10] K.D. Do and J. Pan, Nonlinear formation control of unicycle-type mobile robots, *Robotics and Autonomous Systems*, In Press, Available online 1 November 2006.

[11] H. G. Tanner and A. Kumar, Towards Decentralization of Multi-Robot Navigation Functions, *Proceedings of IEEE International Conference on Robotics and Automation*, Barcelona, pp. 4143-4148, 2005.

[12] H. G. Tanner and A. Kumar, Formation Stabilization of Multiple Agents Using Decentralized Navigation Functions, *Robotics: Science and Systems* I, S. Thrun, G. Sukhatme, S. Schaal and O. Brock (eds), MIT Press, pp. 49Ű56, 2005.

[13] E. Rimon and D. E. Koditschek, Robot Navigation Functions on Manifolds with Boundary, *Advances in Applied Mathematics*, vol. 11, pp. 412-442, 1990.

[14] V. Gazi and K. M. Passino, A Class of Attraction/Repulsion Functions for Stable Swarm Aggregations, *International Journal of Control*, vol. 77, pp. 1567-1579, 2004.

[15] H. G. Tanner, A. Jadbabaie and G. J. Pappas, Stable Flocking of Mobile Agents, Part II: Dynamics Topology, *Proceedings of IEEE Conference on Decision and Control*, Hawaii, pp. 2016-2021, 2003.

[16] S. S. Ge and Y. J. Cui, New Potential Functions For Mobile Robot Path Planning, *IEEE Transactions on Robotics and Automation*, vol. 16, pp. 615-620, 2000.

[17] P. Ogren, M. Egerstedt, and X. Hu, A Control Lyapunov Function Approach to Multi-agent Coordination, *IEEE Transactions on Robotics and Automatation*, vol. 18, pp. 847-851, 2002.

[18] P. Ogren, E. Fiorelli and N. E. Leonard, Cooperative Control of Mobile Sensor Networks: Adaptive Gradient Climbing in a Distributed Environment, *IEEE Transactions on Automatic Control*, vol. 49, No. 8, pp. 1292-1302.

[19] H. Khalil, *Nonlinear systems*, Prentice Hall, 2002.

[20] A. Wells, *Theory and Problems of Lagrangian Dynamics*, New York, 1967.

[21] D. Liberzon, *Switching in Systems and Control*, Birkauser, 2003.

K.D. Do
School of Mechanical Engineering
The University of Western Australia
35 Stirling Highway, Crawley, WA 6009, Australia
E-mail: duc@mech.uwa.edu.au

Received: December 26, 2006

International Journal of Computers, Communications & Control
Vol. II (2007), No. 4, pp. 355-366

Robust Predictive Control using a GOBF Model for MISO Systems

Ali Douik, Jalel Ghabi, Hassani Messaoud

Abstract: In this paper we develop a new method for robust predictive control for MISO systems represented on the Generalized Orthonormal Basis Functions. Unknown But Bounded Error approaches are used to update the uncertainty domain of the resultant model coefficients. This method uses a worst case strategy solved by a min-max optimization problem taking into account the constraints relative to parameter uncertainties and to measurement signals.
Keywords: Predictive Control, Robust, Generalized Orthonormal Basis Functions, MISO, UBBE.

1 Introduction

There has been interest in the use of orthogonal basis functions for the purposes of Robust Model Predictive Control (RMPC) [1, 2, 3, 4]. The most common model structure employing these bases is the well known FIR one. However, the number of terms in the series expansion is high, and this may lead to poor accuracy in the estimated uncertainty domain parameter as well as the control strategy. Another approach is to use Laguerre or Kautz models that are more suitable to represent systems having near or oscillating dynamics [5, 6]. Moreover, using the popular ARMAX model structure [7] involves a small number of parameters but the criterion to be minimized is not convex which may complicate the optimization problem. This paper is a contribution overlapping these methods by developing a new RMPC algorithm for a MISO system represented on the Generalized Orthonormal Basis Functions (GOBF) [8, 9]. However, the main features of using GOBF model in RMPC methods is that the common FIR, Laguerre and Kautz model structures are special cases of this complete construction [10, 11, 12], it is not sensitive to sampling interval choice, it doesn't requires a prior knowledge of the system delay and it operates on a small number of parameters. Furthermore, the criterion is convex on the uncertainty domain of the GOBF model coefficients. The uncertainty domain is determined with Unknown But Bounded Error approaches (UBBE) that updates polytopes, orthotopes, parallelotopes, ellipsoids or limited complexity polytopes [13, 14, 15]. The optimal poles of these basis functions are estimated using a new technique of poles estimation [16, 17].

The paper is organized as follows: In section 2 we present the state space model for the MISO system represented on the GOBF. The predictor output is expressed in section 3. In section 4, robust predictive control method is detailed and the main results are developed. Simulation examples are in section 5 and finally, some conclusions are given in section 6.

2 State-Space Model

This paper considers a MISO system having m input sequences $\{u_1(k), u_2(k), \cdots, u_m(k)\}$ and an output sequence $\{y(k)\}$ that are related according to:

$$y(k) = \sum_{j=1}^{m} G_j(q^{-1}) u_j(k) + e(k) \tag{1}$$

where q^{-1} is the backward shift ($q^{-1}u_j(k) = u_j(k-1)$). $\left\{G_j(q^{-1})\right\}$ describe the unknown system dynamics (assumed stable) and $e(k)$ is the model uncertainty.

The discrete time state-space model for a MISO system represented on the GOBF is defined by:

$$\begin{cases} x(k+1) = Ax(k) + Bu(k) \\ \hat{y}(k) = \theta^T x(k) \end{cases} \tag{2}$$

with:
$u(k) \in \Re^m$ and $\hat{y}(k)$ are the input signal vector and the model output respectively. $x(k)$ is an N dimensional state vector of elements $\left\{x_n^j(k)\right\}_{n=0,1,\cdots,N_j}^{j=1,2,\cdots,m}$ defined by:

$$x_n^j(k) = \mathcal{Z}^{-1}\left\{\mathcal{B}_n^j(z,\underline{\xi_j})\right\} u_j(k) \tag{3}$$

where $\mathcal{Z}^{-1}$ is the inverse transform of z. N_j and $\underline{\xi_j}$ are the truncating order and the poles vector respectively for the j-network of the GOBF. $N = \sum_{j=1}^{m} (N_j + 1)$ is the number of the GOBF model parameter for the MISO system and $\left\{\mathcal{B}_n^j(z,\underline{\xi_j})\right\}_{n=0,1,\cdots,N_j}^{j=1,2,\cdots,m}$ is the GOBF expression given by:

$$\mathcal{B}_n^j(z) = \frac{\sqrt{1-\left|\xi_n^j\right|^2}}{z-\xi_n^j} \prod_{k=0}^{n-1} \left(\frac{1-\bar{\xi}_k^j z}{z-\xi_k^j}\right) \tag{4}$$

where ξ_k^j and its conjugate $\bar{\xi}_k^j$ are the poles for the k-filter of the GOBF.
$\theta \in \Re^N$ is the parameter vector. A and B are $(N \times N)$ and $(N \times m)$ dimensional matrices respectively defined by:

$$A = diag\left(A_j\right)_{j=1,2,\cdots,m}, \quad B = diag\left(B_j\right)_{j=1,2,\cdots,m} \tag{5}$$

where the $(1+N_j) \times (1+N_j)$ dimensional matrix A_j and the $(1+N_j)$ dimensional vector B_j are given by:

$$A_j(a,b) = \begin{cases} \xi_{a-1}^j & \text{if} \quad a = b, \\ F_j(a,b) & \text{if} \quad a \succ b, \\ 0 & \text{if} \quad a \prec b. \end{cases} \tag{6}$$

$$F_j(a,b) = (-1)^{a+b+1} \alpha_{a-1}^j (1 - \xi_{b-1}^j \bar{\xi}_{b-1}^j) \prod_{\ell=b+1}^{a-1} \alpha_{\ell-1}^j \bar{\xi}_{\ell-1}^j \tag{7}$$

$$B_j(b) = (-1)^{b+1} \alpha_{b-1}^j \prod_{\ell=1}^{b-1} \alpha_{\ell-1}^j \bar{\xi}_{\ell-1}^j \qquad (b = 1, \cdots, N_j + 1) \tag{8}$$

And we assume:

$$\alpha_\ell^j = \frac{\sqrt{1-\left|\xi_\ell^j\right|^2}}{\sqrt{1-\left|\xi_{\ell-1}^j\right|^2}}, \qquad \alpha_0^j = \sqrt{1-\left|\xi_0^j\right|^2} \tag{9}$$

3 Step-Ahead Predictor

Equation system (2) can be written in incremental form as:

$$\delta x(k+1) = A\delta x(k) + B\delta u(k) \tag{10}$$

$$\hat{y}(k) = \hat{y}(k-1) + \theta^T \delta x(k) \tag{11}$$

where:

$$\delta u(k) = u(k) - u(k-1), \quad \delta x(k) = x(k) - x(k-1) \tag{12}$$

When the error on the GOBF model is unknown but bounded, the Fourier coefficients are defined by uncertainty intervals. Equation (11) can be then rewritten as:

$$\hat{y}(k) = \hat{y}(k-1) + \theta^T(\varepsilon)\delta x(k) \tag{13}$$

where $\varepsilon \in \Omega$ is the vector of parameter uncertainties and Ω the parameter uncertainty domain. From (13), the p-step ahead predictor can be written as:

$$\hat{y}(k+p/k) = \hat{y}(k+p-1/k) + \theta^T(\varepsilon)\delta x(k+p); \qquad p \geq 1 \tag{14}$$

Using (10) and by successive substitutions we can write:

$$\delta x(k+p) = A^p \delta x(k) + \sum_{q=1}^{p} A^{p-q} B \delta u(k+p-1) \tag{15}$$

Thus, by successive substitution of (15) into (14) we finally have:

$$\hat{y}(k+p/k) = \hat{y}(k) + \theta^T(\varepsilon)\left[K_p - I_N\right]\delta x(k) + \theta^T(\varepsilon)\sum_{q=1}^{p} K_{p-q} B \delta u(k+q-1) \tag{16}$$

where I_N is the identity matrix and K_p is an $(N \times N)$ dimensional matrix defined by:

$$K_p = \begin{cases} \sum_{q=0}^{p} A^q & \text{for } p \geq 0 \\ 0 & \text{for } p \prec 0 \end{cases} \tag{17}$$

The p-step ahead predictor can be written as a sum of two components: the free part and the forced part:

$$\hat{y}(k+p/k) = \hat{y}_l(k+p/k) + \hat{y}_f(k+p/k) \tag{18}$$

with:

$$\hat{y}_l(k+p/k) = \hat{y}(k) + \theta^T(\varepsilon)\left[K_p - I_N\right]\delta x(k) \tag{19}$$

$$\hat{y}_f(k+p/k) = \theta^T(\varepsilon)\sum_{q=1}^{p} K_{p-q} B \delta u(k+q-1) \tag{20}$$

We note by h_1, h_2 and h_u $(h_u \prec h_2)$ the output prediction horizons and the control horizon successively. We assume that $h_1 = 1$. On the prediction horizon $[k+1, k+h_2]$, (18) can be written in matrix form as:

$$\hat{Y}(k,\varepsilon) = \hat{Y}_f(k,\varepsilon) + \hat{Y}_l(k,\varepsilon) \tag{21}$$

where $\hat{Y}(k,\varepsilon)$ is the predictor vector of dimension h_2 defined by:

$$\hat{Y}(k,\varepsilon) = \begin{bmatrix} \hat{y}(k+1/k,\varepsilon) \\ \vdots \\ \hat{y}(k+h_2/k,\varepsilon) \end{bmatrix} \tag{22}$$

The vectors $\hat{Y}_l(k)$ and $\hat{Y}_f(k)$ can be computed using (19) and (20) respectively for $(p = 1, 2, \cdots, h_2)$. Thus, we can write:

$$\hat{Y}_f(k,\varepsilon) = G(\varepsilon)\delta U(k) \tag{23}$$

with:
$\delta U(k)$ is the control increment vector of dimension (mh_u) defined by:

$$\delta U(k) = \begin{bmatrix} \delta u(k) \\ \delta u(k+1) \\ \vdots \\ \delta u(k+h_u-1) \end{bmatrix} \tag{24}$$

where $\delta u(k+p)$ represent the control increment vector defined by:

$$\delta u(k+p) = u(k+p) - u(k+p-1) \qquad \forall \quad p \in [0, h_u - 1] \tag{25}$$

$$u(k+p) = \sum_{q=0}^{p} \delta u(k+p-q) + u(k-1) \tag{26}$$

$G(\varepsilon)$ is an $h_2 \times (mh_u)$ dimensional matrix that represents the impulse response coefficients and defined by:

$$G(\varepsilon) = \begin{bmatrix} G_1(\varepsilon) & 0 & \cdots & 0 \\ G_2(\varepsilon) & G_1(\varepsilon) & \cdots & \vdots \\ \vdots & \vdots & \ddots & \vdots \\ G_{h_u}(\varepsilon) & \cdots & \cdots & G_1(\varepsilon) \\ \vdots & \vdots & \ddots & \vdots \\ G_{h_2}(\varepsilon) & \cdots & \cdots & G_{h_2-h_u+1}(\varepsilon) \end{bmatrix} \tag{27}$$

with $G_p^T(\varepsilon)$ is a vector of dimension m given by:

$$G_p(\varepsilon) = \theta^T(\varepsilon) K_{p-1} B = \sum_{q=1}^{p} \theta^T(\varepsilon) A^{q-1} B \qquad (p = 1, 2, \cdots, h_2) \tag{28}$$

4 Robust Predictive Control Algorithm

4.1 Constraints

The constraints are resulting from uncertainties on the GOBF model coefficients and bounds on control signals and control increments over the control horizon h_u.

$$u_{\min} \leq u(k+p) \leq u_{\max} \qquad \forall \quad p \in [0, h_u - 1] \tag{29}$$

$$\delta u_{\min} \leq \delta u(k+p) \leq \delta u_{\max} \qquad \forall \quad p \in [0, h_u - 1] \tag{30}$$

where:

$$u_{max} = \begin{bmatrix} u_{1\max} \\ \vdots \\ u_{m\max} \end{bmatrix}, \quad u_{min} = \begin{bmatrix} u_{1\min} \\ \vdots \\ u_{m\min} \end{bmatrix} \tag{31}$$

$$\delta u_{max} = \begin{bmatrix} \delta u_{1\max} \\ \vdots \\ \delta u_{m\max} \end{bmatrix}, \quad \delta u_{min} = \begin{bmatrix} \delta u_{1\min} \\ \vdots \\ \delta u_{m\min} \end{bmatrix} \tag{32}$$

Using (26), (29) and (30) we define the set $\delta\Psi$ of constraints on control signals as follows:

$$\delta\Psi = \{\delta U / \Gamma \delta U \leq V\} \tag{33}$$

with Γ is an $(4mh_u) \times (mh_u)$ dimensional matrix and V a vector of dimension $(4mh_u)$.

$$\Gamma = \begin{bmatrix} I_{mh_u} \\ -I_{mh_u} \\ \Delta \\ -\Delta \end{bmatrix}, \quad V = \begin{bmatrix} \delta U_{Max} \\ -\delta U_{Min} \\ U_{Max} - \varphi \\ -U_{Min} + \varphi \end{bmatrix} \tag{34}$$

where I_{mh_u} is the (mh_u) dimensional identity matrix.
The matrix Δ of dimension $(mh_u) \times (mh_u)$ and the vector φ of dimension (mh_u) are given by:

$$\Delta = \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 1 & 1 & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 1 & \cdots & 1 & 1 \end{bmatrix}, \quad \varphi(k-1) = \begin{bmatrix} u(k-1) \\ \vdots \\ u(k-1) \end{bmatrix} \tag{35}$$

U_{Max}, U_{Min}, δU_{Max} and δU_{Min} are (mh_u) dimensional vectors defined as:

$$U_{Max} = \begin{bmatrix} u_{\max} \\ \vdots \\ u_{\max} \end{bmatrix}, \quad U_{Min} = \begin{bmatrix} u_{\min} \\ \vdots \\ u_{\min} \end{bmatrix} \tag{36}$$

$$\delta U_{Max} = \begin{bmatrix} \delta u_{\max} \\ \vdots \\ \delta u_{\max} \end{bmatrix}, \quad \delta U_{Min} = \begin{bmatrix} \delta u_{\min} \\ \vdots \\ \delta u_{\min} \end{bmatrix} \tag{37}$$

4.2 Optimization Criterion

The robust predictive control algorithm using an uncertainty model, is based on a worst case strategy that consists to resolve a min-max optimization problem given by:

$$\min_{\delta U \in \delta\Psi} \max_{\varepsilon \in \Omega} J(\delta U, \varepsilon) \tag{38}$$

The quadratic criterion to be minimized is defined by:

$$J(\delta U, \varepsilon) = \sum_{p=1}^{h_2} \left(\hat{y}(k+p) - r(k+p)\right)^2 + \sum_{j=1}^{m} \left\{ \sum_{p=0}^{h_u-1} \lambda_j^p \delta u_j^2(k+p) \right\} \tag{39}$$

with:

$$\delta u(k+p) = 0 \qquad \text{for} \qquad p \geq h_u \tag{40}$$

where $\lambda_j^p \succ 0 \quad (j = 1, 2, \cdots, m)$ is a weighting factor generally considered constant and equals to λ_j.
$r(k+p)$ represent the reference signal defined on the prediction horizon $[k+1, k+h_2]$.
The quadratic criterion $J(\delta U, \varepsilon)$ can be written in matrix form as:

$$J(\delta U, \varepsilon) = \left\| \hat{Y}(k, \varepsilon) - R(k) \right\|^2 + \left\| \Lambda^{1/2} \delta U(k) \right\|^2 \tag{41}$$

>From (41), we can write:

$$J(\delta U, \varepsilon) = \left(\hat{Y}(k, \varepsilon) - R(k)\right)^T \left(\hat{Y}(k, \varepsilon) - R(k)\right) + \delta U^T(k) \Lambda \delta U(k) \tag{42}$$

where $R(k)$ is an h_2 dimensional reference vector defined by:

$$R(k) = \begin{bmatrix} r(k+1) \\ \vdots \\ r(k+h_2) \end{bmatrix} \tag{43}$$

Λ is an $(mh_u \times mh_u)$ dimensional weighting diagonal matrix defined by:

$$\begin{array}{l} \Lambda = diag(\Lambda_0, \Lambda_1, \cdots, \Lambda_{h_u-1}) \\ \Lambda_p = diag(\lambda_1, \lambda_2, \cdots, \lambda_m); \qquad p = 0, \cdots, h_u - 1 \end{array} \tag{44}$$

Using (21), the matrix form (42) can be rewritten as:

$$J(\delta U, \varepsilon) = \delta U^T \phi(\varepsilon) \delta U + 2\rho^T(\varepsilon)\delta U + \beta(\varepsilon) \tag{45}$$

where ϕ is an $(mh_u \times mh_u)$ dimensional positive definite matrix:

$$\phi(\varepsilon) = G^T(\varepsilon)G(\varepsilon) + \Lambda \tag{46}$$

ρ is a vector of dimension (mh_u):

$$\rho(\varepsilon) = G^T(\varepsilon)\left[\hat{Y}_l(k,\varepsilon) - R(k)\right] \tag{47}$$

β is a scalar defined as follows:

$$\beta(\varepsilon) = \left[\hat{Y}_l(k,\varepsilon) - R(k)\right]^T \left[\hat{Y}_l(k,\varepsilon) - R(k)\right] \tag{48}$$

Since the criterion is convex over the parameter uncertainty set, the maximization problem over this set can be reduced to the maximization over its vertices. When the parameter set is an ellipsoid, it is approximated by the orthotope containing it. Therefore the optimization problem (38) becomes:

$$\min_{\delta U \in \delta\Psi} \max_{\varepsilon \in S} J(\delta U, \varepsilon) \tag{49}$$

where S is the set of vertices of the orthotope. The number of constraints is given by:

$$L = 2^N + 4mh_u \tag{50}$$

where 2^N is the number of the vertices of the domain S for the MISO system.
The RMPC algorithm using a GOBF model for a MISO system can be summarized as follow:

- compute the matrices A and B from (5),
- determine the set of vertices,
- select the parameters h_2 and h_u,
- select the weighting matrix coefficients,
- compute the matrices $K_p \quad (p = 1, \cdots, h_2)$ from (17),
- compute the coefficients $G_p \quad (p = 1, \cdots, h_2)$ from (28),
- compute the references.

Computation at each sampling period:

- compute the free component $\hat{Y}_l(k)$ using (19),
- compute the quadratic criterion using (45),
- determine the control increment vector using (49).

5 Simulation Examples

In this section we will illustrate the utility of the robust predictive control method by presenting some simulation examples. To begin with, suppose we have a MISO system with $m = 2$ input sequences and a number of $H = 300$ point data record generated by the following model:

$$y(k) = \frac{0.102z^{-1} - 0.751z^{-2}}{1 - 0.745z^{-1}}u_1(k) + \frac{-(0.152z^{-1} + 0.255z^{-2})}{(1 + 0.7047z^{-1})(1 - 0.3547z^{-1})}u_2(k) + e(k) \tag{51}$$

where $u_1(k), u_2(k)$, $y(k)$ and $e(k)$ are the inputs, the output and the model error respectively. The model error is assumed to be bounded such $|e(k)| \leq 4.51$ and the input signals are uniformly distributed sequences. In this simulation we approximate this model by the GOBF model where the truncating order and the optimal poles are: $N_{opt} = 4$; $\underline{\xi_{opt}} = (0.7450 \;\; 0 \;\; 0.3547 \;\; -0.7047)$. The process output and the GOBF model output are illustrated in figure 1.

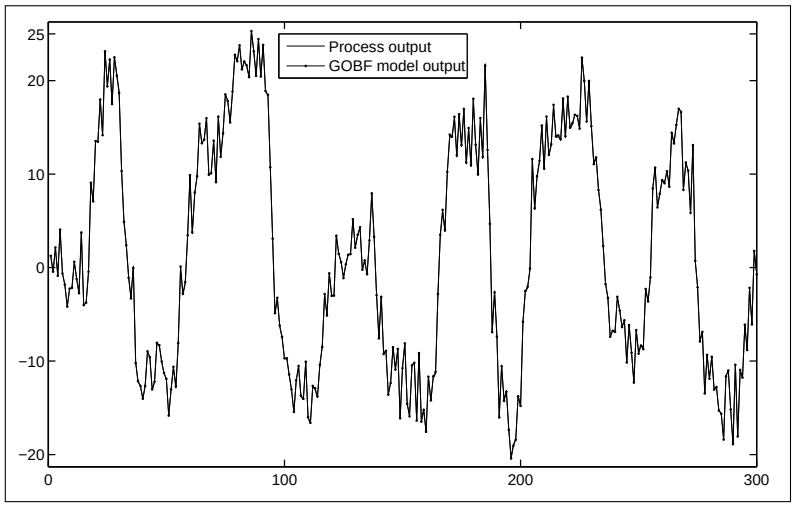


Figure 1: Process output and GOBF model output

The center and uncertainty intervals (UI) of the ellipsoid are given in table 1. The tuning parameters used in this simulation are: $h_2 = 8, h_u = 2, \lambda_1 = 1$ and $\lambda_2 = 1$.

Table 1: Ellipsoid Performances

Ellipsoidal center	-0.6326	-0.9135	-0.2266	-0.1260
Uncertainty intervals	0.3797	0.9320	0.7085	1.9076

To validate the control method we plot in figure 2 the GOBF model output and the reference signal. The control signals and the control increment signals are illustrated in figure 3 and 4 successively. The picks of the control signals as well as the control increment signals are due to the changed reference signal from -40 to +40 at the iterations 100 and 200. Therefore, we notice the rapid convergence of the model output to the reference signal. This is predictable since we optimize a tracking criterion. Other simulation examples with different GOBF models and reference signals have been studied and yielded the same results.

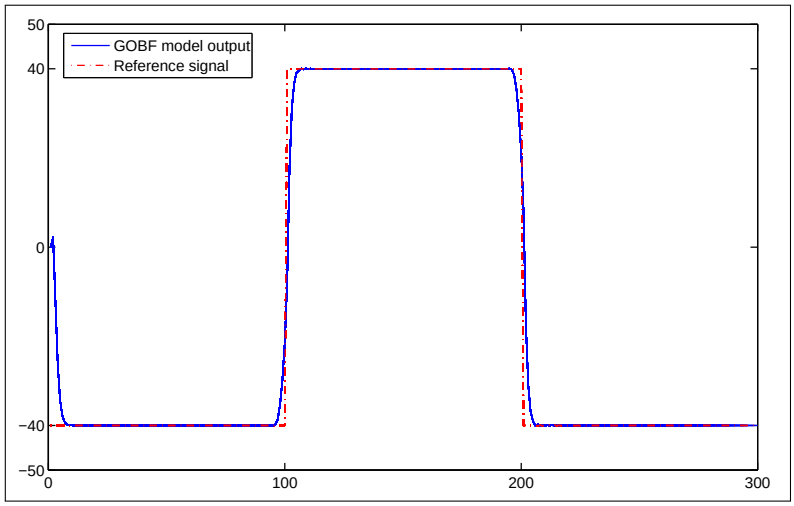


Figure 2: Reference signal and GOBF model output

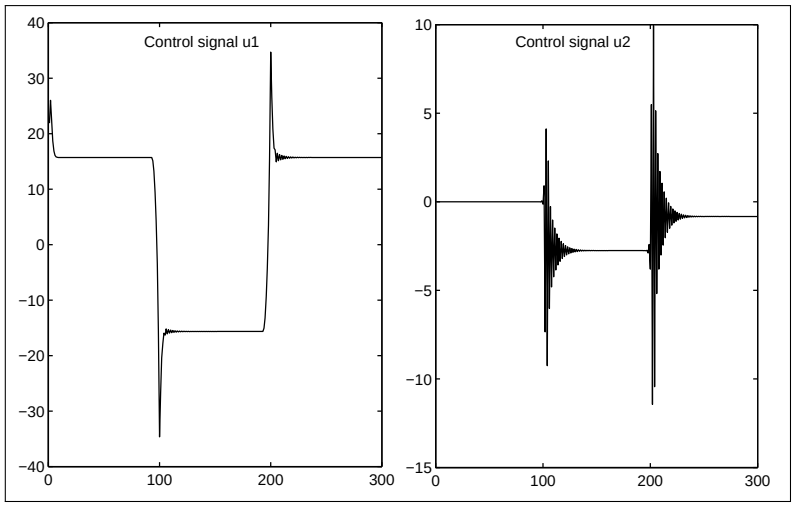


Figure 3: Control signals

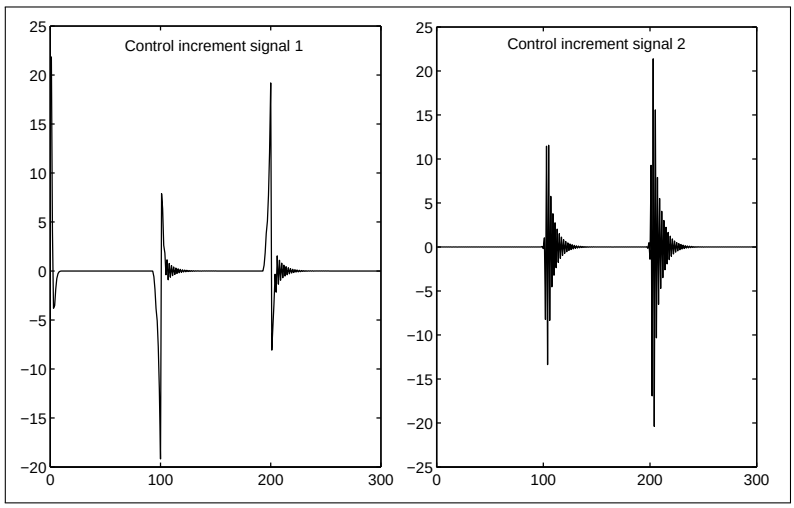


Figure 4: Control increment signals

On the other hand, the influence of the error bounds on the GOBF model output in the case of an ellipsoid domain is studied by considering 3 different SNR (signal to noise ratio). The table 2 gives the centers and the uncertainly intervals where the figure 5 illustrates the model outputs and the reference signal fixed arbitrary. This figure shows the similar convergence of the model outputs to the reference signal. Thus, we conclude that for different error bounds, we obtain the same GOBF model output. The control method has been tested with different reference signals and error bounds that yielded the same results.

Finally we study the influence of different uncertainty domains such an ellipsoid, an orthotope and a polytope. The table 3 regroups the centers and the uncertainty intervals of these domains. The model outputs correspondent are shown in figure 6. By examining this figure we notice that the model outputs converge simultaneously to the reference signal. So, we conclude that the type of the parameter domain has no influence on this control method. Other experiences with different reference signals and domain parameter have been realized and yielded the same results.

Table 2: Ellipsoid performances for different error bounds

SNR=5	**Center**	-0.5698	-1.0975	-0.2557	-0.0844
	UI	0.7915	1.9507	1.4634	3.9237
SNR=10	**Center**	-0.6071	-0.9886	-0.2377	-0.1086
	UI	0.5517	1.3550	1.0246	2.7549
SNR=20	**Center**	-0.6326	-0.9135	-0.2266	-0.1260
	UI	0.3797	0.9320	0.7085	1.9076

Table 3: Domain performances (SNR=20)

Ellipsoid	**Center**	-0.6326	-0.9135	-0.2266	-0.1260
	UI	0.3797	0.9320	0.7085	1.9076
Orthotope	**Center**	-0.6950	-0.7551	0.1082	-0.1356
	UI	0.6403	1.6896	1.7069	4.2133
Polytope	**Center**	-0.6924	-0.7556	-0.1968	-0.1754
	UI	0.0236	0.0307	0.0472	0.1095

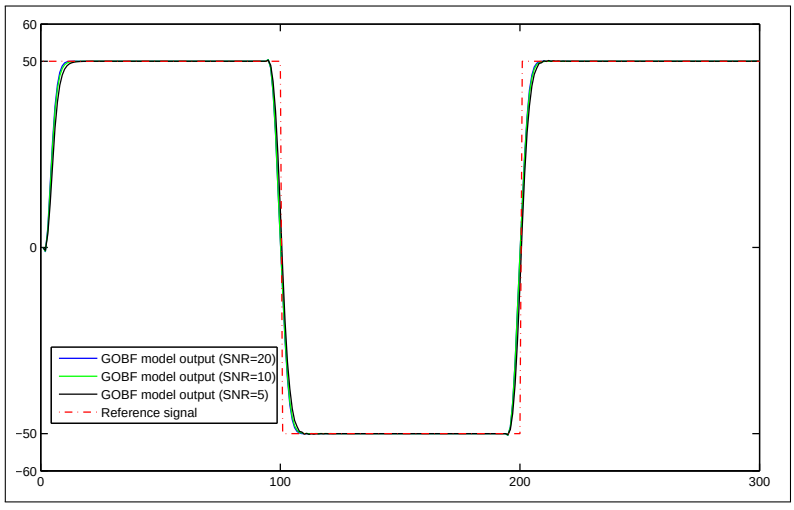


Figure 5: Model outputs for 3 different SNR of an ellipsoid domain

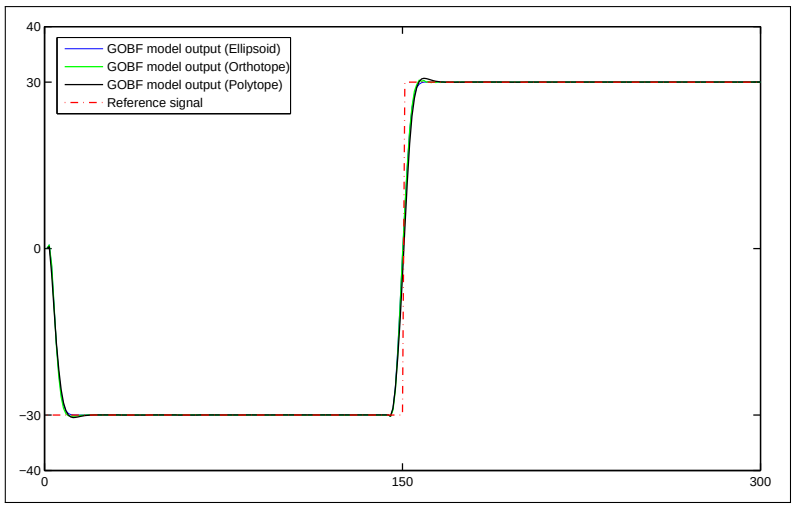


Figure 6: Model outputs for different uncertainty domains

6 Conclusion

This paper has presented a new robust predictive control method based on the GOBF model for a MISO system. A min-max problem is solved taking into account the uncertainties on the model coefficients and the constraints on the control signals. The uncertainty parameter domain can be an ellipsoid, an orthotope or a polytope and the performance criterion is optimized with respect to constraints relative to parameter uncertainties and measurement constraints. The implication of these results in the context of system controls is that the GOBF can be used to deliver state space models suitable to synthesize a robust predictive control without affecting the computational complexity and the performance of the method. Finally, it should also be noted that this control method provides best results and may be synthesized for a MIMO system represented on the GOBF.

Bibliography

[1] G. Oliveira, G. Favier, G. Dumont, and W. Amara, Predictive Controller Based on Laguerre Filters Modeling, *In Proc. 13th IFAC World Congress*, San Francisco, USA, vol. G, pp. 375-380, 1996.

[2] A. Mbarek, H. Messaoud and G. Favier, Robust predictive control using Kautz model, *In Proc. 10th IEEE International Conf. Electronics, Circuits and Systems*, pp. 184-187, December 2003.

[3] L.P. Wang, Discrete time model predictive control design using Laguerre functions, *Journal of Process Control*, 14:131-142, 2004.

[4] E.M. El Adel, M. Ouladsine, and L. Radouane, Predictive steering control using Laguerre series representation, *In Proc. 2003 American Control Conf.*, vol.1, pp. 439-445, June 2003.

[5] B. Wahlberg, System identification using Laguerre models, *IEEE Trans. on Automatic Control*, 36(5): 551-562, 1991.

[6] B. Wahlberg, System identification using Kautz models, *IEEE Trans. on Automatic Control*, 39(6):1276-1282, 1994.

[7] A. Gutierrez and E. Camacho, Robust Adaptive Control for Processes with Bounded Uncertainties, *In Proc. 3rd ECC*, Roma, vol.2, pp. 1295-1300, 1995.

[8] J. Ghabi, A. Douik and H. Messaoud, A New Modelling Approach of MIMO linear Systems using the generalized Orthonormal basis functions, *In Proc. of the 6th ISPRA'07*, Greece, pp. 192-196, February 2007.

[9] J. Ghabi, A. Douik and H. Messaoud, New Methods of Modelling and Parameter Estimation for MIMO linear Systems using Generalized Orthonormal Basis Functions, *Journal on Systems and Control*, vol. 2, pp. 133-140, 2007.

[10] B. Ninness and F. Gustafsson, A unifying construction of orthonormal bases for system identification, *IEEE Trans. on Automatic Control*, 42 (4): 515-521, 1997.

[11] P. S. C. Heuberger, P. M. J. Van den Hof, and O. H. Bosgra, A generalized orthonormal basis for linear dynamical systems, *IEEE Trans. on Automatic Control*, 40(3):451-465, 1995.

[12] Paul Van den Hof, Brett Ninness, System Identification with Generalized Orthonormal Basis Functions, *In Modelling and Identification with Rational Orthogonal Basis Functions, Springer Verlag*, chap.4, pp. 61-102, 2005.

[13] G. Favier and L. Arruda, Review and Comparison of Ellipsoidal Bounding Algorithms, *In M. et al., editor, Bounding Approaches to system identification, Plenum Press*, New York, chap.4, pp. 43-68, 1996.

[14] H. Messaoud and G. Favier, Recursive determination of parameter uncertainty intervals for linear models with unknown but bounded errors, *In Proc. 10th IFAC Symposium on System Identification*, Copenhagen, Denmark, pp. 365-370, 1994.

[15] S. Maraoui and H. Messaoud, Design and comparative study of limited complexity bounding error identification algorithms, *IFAC, Symposium on System Structure and Control*, Prague, Cheque Republique, pp. 29-31, 2001.

[16] J. Ghabi, A. Douik and H. Messaoud, A New Estimation Method of the Poles for the generalized Orthonormal bases filters, *In Proc. of the 6th ISPRA'07*, Greece, pp. 197-202, February 2007.

[17] J. Ghabi, A. Douik and H. Messaoud, A New Technique of Poles Estimation for Generalized Orthonormal Basis Functions, *Journal on Systems and Control*, vol.2, pp. 125-132, 2007.

Ali Douik, Jalel Ghabi and Hassani Messaoud
Ecole Nationale d'Ingénieurs de Monastir (ENIM)
Département de Génie Electrique
Laboratoire ATSI
Rue Ibn El Jazzar, 5019 Monastir Tunisie
E-mail: Ali.douik@enim.rnu.tn, jalel.ghabi@yahoo.fr, hassani.messaoud@enim.rnu.tn

Received: June 30, 2007

Ali Douik was born in Tunis, Tunisia. He received the Master degree in 1990 and the Ph.D. degree in 1996 of science Electrical Genius from the "Ecole Normale Supérieur de l'Enseignement Technique de Tunis". He is currently "Maitre assistant"in the "Ecole Nationale d'Ingénieurs de Monastir"and Director of the "Département de Génie Electrique". His research is related to Signal Processing and Automatic Control.

Jalel Ghabi was born in Kairouan, Tunisia. He received the diploma of Graduate Engineer in Electrical Genius from the "Ecole Nationale d'Ingénieurs de Monastir", in 1997. He obtained her Master of Automatic and Industrial Data Processing from "Ecole Nationale d'Ingénieurs de Sfax", in 2003. He is currently preparing the PhD. Degree in Control Automatic in the laboratory ATSI "Automatique, Traitement de Signal et Imagerie". His research is related to the Robust Predictive Control of MIMO systems using Generalized Orthonormal Basis Functions.

Hassani Messaoud was born in Mahdia, Tunisia. He received the Master degree from the "Ecole Normale Supérieur de l'Enseignement Technique de Tunis", in 1985 and the Ph.D. degree in Automatic Control from the "Ecole Centrale de Lille, France", in 1993. In 2001, he received the ability degree from the "Ecole Nationale d'Ingénieurs de Tunis". He is presently Professor in the "Ecole Nationale d'Ingénieurs de Monastir"and Director of the laboratory ATSI. He has been the supervisor of several PhD thesis and is author or co-author of several Journal articles. His research is related to Automatic Control and Signal Processing.

International Journal of Computers, Communications & Control
Vol. II (2007), No. 4, pp. 367-374

An Efficient Numerical Integration Algorithm for Cellular Neural Network Based Hole-Filler Template Design

V. Murugesh, Krishnan Batri

Abstract: This paper presents, a design method for the template of the hole-filler used to improve the performance of the character recognition using Numerical integration algorithms. This is done by analyzing the features of the hole-filler template and the dynamic process of CNN and by using popular numerical algorithms to obtain a set of inequalities satisfying its output characteristics as well as the parameter range of the hole-filler template. Some simulation results and comparisons are also presented.
Keywords: Cellular Neural Networks; Euler Algorithm; RK-Gill Algorithm; RK-Butcher Algorithm; Ordinary differential equations, Hole-filler.

1 Introduction

Cellular Neural Networks (CNNs) are analog, time-continuous, nonlinear dynamical systems and formally belong to the class of recurrent neural networks. Since their introduction in 1988 (by Chua and Yang [1, 2], it has been the subject of intense research. Initial applications include image processing, signal processing, pattern recognition and solving partial differential equations, etc.

Runge-Kutta (RK) methods have become very popular, both as a computational technique as well as a subject of research, which are discussed by Butcher [3, 4]. This method was derived by Runge around the year 1894 and extended by Kutta a few years later. They developed algorithms to solve differential equations efficiently and these are the equivalent of approximating the exact solutions by matching 'n' terms of the Taylor series expansion.

Butcher [3] derived the best RK pair along with an error estimate and by all statistical measures it appeared as the RK-Butcher algorithm. This RK-Butcher algorithm is nominally considered as sixth order, since it requires six function evaluations, but in actual practice the "working order" is close to five (fifth order).

Bader [4, 5] introduced the RK-Butcher algorithm for finding the truncation error estimates and intrinsic accuracies and the early detection of stiffness in coupled differential equations that arise in theoretical chemistry problems. Recently Devarajan et al [7] used the RK-Butcher algorithm for finding the numerical solution of an industrial robot arm control problem. Oliveria [8] introduced the popular RK-Gill algorithm for the evaluation of 'effectiveness factor' of immobilized enzymes.

In this paper, we describe the dynamic behavior of CNN in section 2, Hole-filler template design ideas in Section 3, Numerical integration algorithms and its description is shown in Section 4, and simulation results in Section 5.

2 Dynamic Analysis of CNN

The dynamic equation of cell $C(i,j)$ in an M x N cellular neural network is given by Chua and Yang [1, 2].

$$C\frac{dx_{ij}(t)}{dt} = -\frac{1}{R_x}x_{ij}(t) + \sum_{C(k,l)\in N_r(i,j)} A(i,j;k,l)Y_{kl}(t) + \sum_{C(k,l)\in N_r(i,j)} B(i,j;k,l)U_{kl} + I \tag{1}$$

$$Y_{ij}(t) = \left[\left|x_{ij}(t)+1\right| - \left|x_{ij}(t)\right| - 1\right], 1 \le i \le M, 1 \le j \le n \tag{2}$$

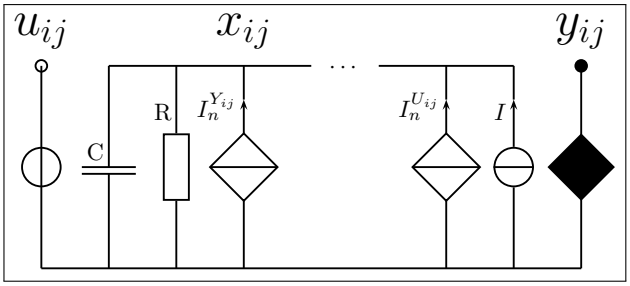


Figure 1: CNN–Cell

wherex_{ij}, y_{ij} and u_{ij} are the state voltage, output voltage and input voltage respectively and they are functions of time t. R_x is a linear resistance, C is a linear capacitor, and $A(i,j;k,l)$ and $B(i,j;k,l)$ are the transconductances of the output and input voltages of $C(k,l)$ with respect to $C(i,j)$ called the cloning templates of CNN. $N_r(i,j)$ denotes the r^{th}- neighbor of $C(i,j)$ and I is an independent current source. From equation (2) one can see that the output voltage is nonlinear. We can rewrite the cell equation (1) as follows:

$$C\frac{dx_{ij}(t)}{dt} = -f\left[x_{ij}(t)\right] + g(t) \tag{3}$$

Where

$$f\left[x_{ij}(t)\right] = \frac{1}{R_x}x_{ij}(t) \tag{4}$$

$$g(t) = \sum_{\substack{C(k,l)\in N_r(i,j)\\ C(k,l)\neq C(i,j)}} A(i,j;k,l)Y_{kl}(t) + \sum_{C(k,l)} B(i,j;k,l)U_{kl} + I \tag{5}$$

3 Hole-filler Template Design

The Hole-Filler is a cellular neural network discussed by Yin et al [9], which fills up all the holes and remains unaltered outside the holes in a bipolar image. Let $R_x = 1$, $C = 1$ and let +1 stand for the black pixel and -1 for the white one. We shall discuss the images having holes enclosed by the black pixels, when the bipolar image is input with $U = \{u_{ij}\}$ into CNN. The initial state values are set as $X_{ij}(0) = 1$. From the equation (2) the output values are $Y_{ij}(0) = 1$, $1 \le i \le M$, $1 \le j \le N$.

Suppose that the template A and B and the independent current source I are given as

$$A = \begin{bmatrix} 0 & a & 0 \\ a & b & a \\ 0 & a & 0 \end{bmatrix}, \quad a > 0, \quad b > 0, \quad B = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 4 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad I = -1 \tag{6}$$

Where the template parameters a and b are to be determined. In order to make the outer edge cells become the inner ones, normally auxiliary cells are added along the outer boundary of the image, and their state values are set to zeros by circuit realization, resulting in the zero output values. The state equation (1) can be rewritten as

$$\frac{dx_{ij}(t)}{dt} = -x_{ij}(t) + \sum_{C(k,l)\in N_r(i,j)} A(i,j;k,l)Y_{ij}(t) + 4u_{ij}(t) - I \tag{7}$$

For the cell $C(i,j)$, we call the cells $C(i+1,j)$, $C(i-1,j)$, $C(i,j+1)$ and $C(i,j-1)$ to be the non-diagonal cells. Here, several cases are to be considered.

Case 1: The input value $u_{ij} = +1$ for cell $C(i, j)$, signaling the black pixel. Because the initial state value of the cell $C(i, j)$ has been set to 1, $x_{ij}(0) = 1$, and from equation (2) its initial output value is also $y_{ij}(0) = 1$. According to the hole-filler demands, its eventual output should be $y_{ij}(\infty) = 1$. To obtain this result we set

$$\frac{dx_{ij}(t)}{dt} \geq 0 \tag{8}$$

Substituting this input $u_{ij} = 1$ and equation (6) into equation (7), we obtain

$$\frac{dx_{ij}(t)}{dt} = -x_{ij}(t) + a\left[y_{(i-1)j}(t) + y_{(i+1)j}(t) + y_{i(j-1)}(t) + y_{i(j+1)}(t)\right] + by_{ij}(t) + 3 \tag{9}$$

Combining equations (8) and (9) and considering the minimum value of $x_{ij}(t) = 1$ this case yields

$$a\left[y_{(i-1)j}(t) + y_{(i+1)j}(t) + y_{i(j-1)}(t) + y_{i(j+1)}(t)\right] + by_{ij}(t) + 2 \geq 0 \tag{10}$$

To facilitate our discussion, two sub cases are distinguished.

Sub Case 1: The cell $C(i, j)$ is inside the holes. Since $x_{ij}(0) = 1$, from equation (2) its initial output value $y_{ij}(0) = 1$. Considering equations (8) and (2), $y_{ij}(t) \geq 1$. According to the hole-filler demands, its initial output of non-diagonal black pixels should not be changed inside the holes. The weights of a and b are equal to +4 and +1, respectively.

Since $A(i, j; k, l) > \frac{1}{R_x}$ the parameter b is found to be $b > 1$, or

$$4a + b + 2 \geq 0, \quad b > 1 \tag{11a}$$

Sub Case 2: The cell $C(i, j)$ is outside the holes. To satisfy equation (10), we need to check only the minimum value on the left-hand side of equation (10). This is true when there are four non-diagonal white pixels around the cell $C(i, j)$, where the weight of a in equation (10) is -4. Since $y_{ij}(t) \geq 1$, the weight of b is equal to 1. Combining this with $b > 1$ gives

$$-4a + b + 2 \geq 0, \quad b > 1 \tag{11b}$$

Case 2: The input value of cell $C(i, j)$ is $u_{ij} = 1$, signaling the white pixel. Substituting this input value in equation (7) gives

$$\frac{dx_{ij}(t)}{dt} = -x_{ij}(t) + a\left[y_{(i-1)j}(t) + y_{(i+1)j}(t) + y_{i(j-1)}(t) + y_{i(j+1)}(t)\right] + by_{ij}(t) - 5 \tag{12}$$

Sub Case 1: The cell $C(i, j)$ is inside the holes. Since $x_{ij}(0) = 1$, from equation (2) its initial output value is $y_{ij}(0) = 1$. According to the hole-filler demands, the holes should be filled by the black pixels, whereas its initial black pixels remain unaltered:

$$\frac{dx_{ij}(t)}{dt} \geq 0 \tag{13}$$

Combining equations (12) and (13) and considering $x_{ij}(t) \geq 1$ yields

$$a\lfloor y_{(i-1)j}(t) + y_{(i+1)j}(t) + y_{i(j-1)}(t) + y_{i(j+1)}(t)\rfloor + by_{ij}(t) - 6 \geq 0 \tag{14}$$

where we use the minimum value of $x_{ij}(t)$ in equation (12). Since the cell is inside the holes, its initial output of non-diagonal black pixels remain unchanged. The weight of a and b are equal to +4 and +1, respectively. Combining this with $b > 1$ gives

$$4a + b - 6 \geq 0, \quad b > 1 \tag{15}$$

Sub Case 2: The cell $C(i,j)$ is outside the holes. Since $x_{ij}(0) = 1$, from equation (2) its initial output value is $y_{ij}(0) = 1$. According to the hole-filler demands, the final output of this cell should be white, and in this case $y_{ij}(\infty) \leq -1$.

$$\frac{dx_{ij}(t)}{dt} < 0 \tag{16}$$

Combining equations (12) and (16) and considering $x_{ij}(t) \leq 1$. we get

$$a\lfloor y_{(i-1)j}(t) + y_{(i+1)j}(t) + y_{i(j-1)}(t) + y_{i(j+1)}(t)\rfloor + by_{ij}(t) - 6 < 0 \tag{17}$$

where we use the maximum value of $x_{ij}(t)$ in equation (12).

Initially $y_{ij}(0) = 1$. How can the output of cell $C(i,j)$ be changed to -1?. Where does this change begin?. First we consider the situation where the change begins from the inside of the bipolar image. If the maximum value on the left-hand side in equation (17) is less than zero, equation (17) holds. Inside the image and outside the holes, the maximum weights of a and b are +4 and +1, respectively. This case was described by equation (15). In fact, the change of the output of the cell $C(i,j)$ is like a wave propagating from the edges to the inside of the image and it is verified from the simulated result. Therefore, we should first consider the edge cell $C(i,j), i = 1$ or $M, j = 1$ or N. For this the maximum weight of a in equation (17) is +3, which is also the maximum weight of a outside the holes. The maximum weight of b is +1, occurring at the initial time:

$$3a + b - 6 < 0, \quad b > 1 \tag{18}$$

Combining Cases 1 and 2, we obtain

$$\begin{aligned} 3a + b - 6 &< 0, \\ 4a + b - 6 &\geq 0, \\ -4a + b + 2 &\geq 0. \end{aligned} \tag{19}$$

4 Numerical Integration Algorithms

The CNN dynamics on a digital architecture requires discretization in time and suitable numerical integration algorithms. Three of the most widely used Numerical Integration Algorithms are used in Raster CNN Simulation described here. They are the Euler's Algorithm, RK-Gill Algorithm discussed by Oliveria [8] and the RK-Butcher Algorithm discussed by Badder [5, 6] and Murugesh and Murugesan [10, 11, 12].

4.1 Euler Algorithm

Euler's method is the simplest of all algorithms for solving ODEs. It is explicit formula which uses the Taylor-series expansion to calculate the approximation.

$$x_{ij}((n+1)\tau) = x_{ij}(\pi\tau) + \tau f'(x(\pi\tau)) \tag{20}$$

4.2 RK-Gill Algorithm

The RK-Gill algorithm discussed by Oliveria [8] is an explicit method requiring the computation of four derivatives per time step. The increase of the state variable x^{ij} is stored in the constant k_1^{ij}. This

result is used in the next iteration for evaluating k_2^{ij}. The same must be done for k_3^{ij} and k_4^{ij}.

$$\begin{aligned} k_1^{ij} &= f'(x_{ij}(\tau\pi)) \\ k_2^{ij} &= f'\left(x_{ij}(\tau\pi)+\frac{1}{2}k_1^{ij}\right) \\ k_3^{ij} &= f'\left(x_{ij}(\tau\pi)+\left(\frac{1}{\sqrt{2}}-\frac{1}{2}\right)k_1^{ij}+\left(1-\frac{1}{\sqrt{2}}\right)k_2^{ij}\right) \\ k_4^{ij} &= f'\left(x_{ij}(\tau\pi)-\frac{1}{\sqrt{2}}k_2^{ij}+\left(1+\frac{1}{\sqrt{2}}\right)k_3^{ij}\right) \end{aligned} \tag{21}$$

The final integration is a weighted sum of the four calculated derivatives:

$$x_{ij}((n+1)\tau) = x_{ij}+\frac{1}{6}\left[kij_1+\left(2-\sqrt{2}\right)k_2^{ij}+\left(2+\sqrt{2}\right)k_3^{ij}+k_4^{ij}\right] \tag{22}$$

4.3 RK-Butcher Algorithm

The RK-Butcher algorithm discussed by Badder [5, 6] and Murugesh and Murugesan [10, 11, 12], is an explicit method. It starts with a simple Euler step. The increase of the state variable x_{ij} is stored in the constant k_1^{ij}. This result is used in the next iteration for evaluating k_2^{ij} . The same must be done for k_3^{ij}, k_4^{ij}, k_5^{ij} and k_6^{ij}.

$$\begin{aligned} k_1^{ij} &= \tau f'(x_{ij}(\pi\tau)) \\ k_2^{ij} &= \tau f'\left(x_{ij}(\pi\tau)+\frac{1}{4}k_1^{ij}\right) \\ k_3^{ij} &= \tau f'\left(x_{ij}(\pi\tau)+\frac{1}{8}k_1^{ij}+\frac{1}{8}k_2^{ij}\right) \\ k_4^{ij} &= \tau f'\left(x_{ij}(\pi\tau)-\frac{1}{2}k_2^{ij}+k_3^{ij}\right) \\ k_5^{ij} &= \tau f'\left(x_{ij}(\pi\tau)+\frac{3}{16}k_1^{ij}+\frac{9}{16}k_4^{ij}\right) \\ k_6^{ij} &= \Delta t f\left(x_{ij}(\pi\tau)-\frac{3}{7}k_1^{ij}+\frac{2}{7}k_2^{ij}+\frac{12}{7}k_3^{ij}-\frac{12}{7}k_4^{ij}+\frac{8}{7}k_5^{ij}\right) \end{aligned} \tag{23}$$

The final integration is a weighted sum of the five calculated derivatives:

$$x_{ij}((n+1)\tau) = \frac{1}{90}\left(7k_1^{ij}+32k_3^{ij}+12k_4^{ij}+32k_5^{ij}+7k_6^{ij}\right) \tag{24}$$

5 Simulated Results

This Hole-filler template has been simulated using Pentium IV Machine with 3.0 Ghz. speed using different Numerical integration algorithms. The Settling time T and integration time T_s is obtained with various step sizes is to be displayed below in the Table-1. The settling time T_s describes the time from start of integration until the last cell leaves the interval [-1.0, 1.0] which is based on certain limit (e.g.,$\left|\frac{dx}{dt} < 0.01\right|$). The simulation shows the desired output for every cell. We use +1 and -1 to indicate the black and white pixels, respectively.

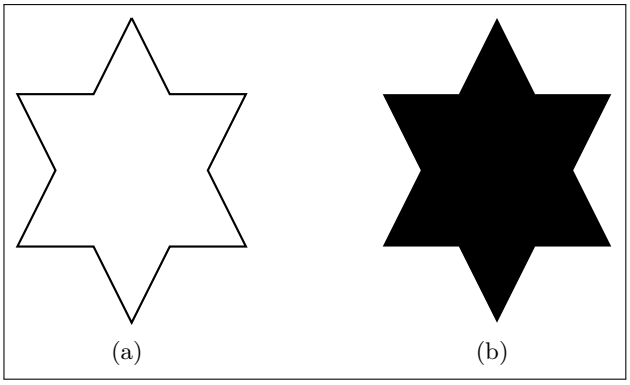


Figure 2: Image Before and After Hole Filling

Step Size	Euler Algorithm		RK-Gill Algorithm		RK-Butcher Algorithm	
	Settling Time(T)	Integration Time(T_s)	Settling Time(T)	Integration Time(T_s)	Settling Time(T)	Integration Time(T_s)
0.5	6.5	2.5	6.8	2.4	5.8	2.4
0.6	15.5	12.7	16.4	13.7	11.4	12.5
0.7	32.5	28.3	32.0	27.4	30.0	27.2
0.8	35.0	30.7	34.6	30.0	32.4	29.6
0.9	36.8	32.6	36.6	32.0	34.2	31.6
1.0	37.9	33.6	37.6	33.0	36.0	32.8
1.5	44.8	36.8	45.7	36.9	41.1	36.0
2.0	47.4	43.2	48.2	43.6	46.2	42.8
2.5	50.6	45.6	52.6	44.5	48.3	44.7
3.0	53.5	49.3	54.8	50.2	52.3	49.2

Table 1: Simulated Results of Hole-Filler Template Design

Example
The templates A, B and I are given as follows:

$$A = \begin{bmatrix} 0 & 1.0 & 0 \\ 1.0 & 3.0 & 1.0 \\ 0 & 1.0 & 0.0 \end{bmatrix}, \quad B = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 4 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad I = -1.0$$

Using the simulation program developed in C++, the input image is shown in Figure-2(a) and the output image in Figure-2(b). The obtained result is represented in Table-1. From the table-1, we find that RK-Butcher algorithm yields less settling time and integration time compared to Euler and RK-Gill algorithms.

6 Conclusion

It is shown that the cellular neural network based hole-filler template could be designed from its dynamic behavior using different numerical algorithms, and also the template for other cellular neural network can similarly be designed. The hole is filled and the outside image remains the same. The templates of the cellular neural network are not unique and this is important in its implementation.

Bibliography

[1] L. O. Chua, L. Yang, Cellular Neural Networks: Theory, *IEEE Transactions on Circuits and Systems*, Vol. 35, pp. 1257 - 1272, 1988.

[2] L. O. Chua, L. Yang, Cellular Neural Networks: Applications, *IEEE Transactions on Circuits and Systems*, Vol. 35, pp. 1273 - 1290, 1988.

[3] J. C. Butcher, *The Numerical Analysis of Ordinary Differential Equations: Runge-Kutta and General Linear Methods*, Chichester: John Wiley, 1987.

[4] J. C. Butcher, *Numerical Methods for Ordinary Differential Equations*, Chichester: John Wiley, 2003.

[5] M. Bader, *A comparative study of new truncation error estimates and intrinsic accuracies of some higher order Runge-Kutta algorithms*, Computers & Chemistry, Vol. 11 pp. 121-124, 1987.

[6] M. Bader, *A new technique for the early detection of stiffness in coupled differential equations and application to standard Runge-Kutta algorithms*, Theoretical Chemistry Accounts, Vol. 99, pp. 215-219, 1988.

[7] G. Devarajan, V. Murugesh, K. Murugesan, *Numerical Solution of Second-order Robot Arm Control Problem using Runge-Kutta Butcher Algorithm*, International Journal of Computer Mathematics, Vol. 83 pp. 345-356, 2006.

[8] S. C. Oliveira, *Evaluation of effectiveness factor of immobilized enzymes using Runge-Kutta-Gill method: how to solve mathematical undetermination at particle center point?*, Bio Process Engineering, Vol. 20, pp. 85-187, 1999.

[9] C. L. Yin, J. L. Wan, H. Lin, W. K. Chen, *The cloning template design of a cellular neural network*, Journal of the Franklin Institute, Vol. 336, pp. 903-909, 1999.

[10] V. Murugesh, K. Murugesan, *Comparison of Numerical Integration Algorithms in Raster CNN Simulation*, Lecture Notes in Computer Science, Vol. 3285 pp. 115-122, 2004.

[11] V. Murugesh, K. Murugesan, *Simulation of Cellular Neural Networks using the RK-Butcher algorithm*, International Journal of Management and Systems, Vol. 21, pp. 65-78, 2005.

[12] V. Murugesh, K. Murugesan, *Simulation of Time-Multiplexing Cellular Neural Networks with Numerical Integration Algorithms*, Lecture Notes in Computer Science, Vol. 3991, pp. 115-122, 2005.

V. Murugesh
Department of Information and Communication Engineering
Hannam University
133 Ojung-dong Daeduk-gu, Daejeon 306-791, Republic of Korea
E-mail: murugesh72@gmail.com

K. Batri
Department of Computer Science and Engineering
Muthyammal Engineering College
Rasipuram 637 408, India
E-mail: krishnan.batri@gmail.com

Received: July 25, 2007

Dr. V. Murugesh obtained his Bachelor of Science in Computer Science and Master of Computer Applications degree from Bharathiar University, Coimbatore, India during 1992 and 1995 respectively. Completed his PhD in Computer Science from Bharathidasan University, Tiruchirappalli, India during 2006. He has held various positions at National Institute of Technology, Tiruchirappalli, India and Sona College of Technology, Salem, India. Currently, he is working as Assistant Professor in the Department of Information and Communication Engineering at Hannam University, Daejeon, Republic of Korea. His fields of interest are in Neural Network based Image Processing and Scientific computing. He has published more than 30 technical papers in International, National journals and conferences.

Krishnan Batri received the M.E. from Madurai Kamarj University in 2003. He is a Research Scholar with the Department of Computer Science and Engineering in the National Institute of Technology Tiruchirapalli, Tamil Nadu, India. Currently he is working as a Assistant Professor with the department of Computer Science and Engineering at Muthayammal Engineering College, Rasipuram, Tamilnadu, India. His research interests include Information Retrieval, Data fusion and Genetic algorithms.

International Journal of Computers, Communications & Control
Vol. II (2007), No. 4, pp. 375-387

Fuzzy and Neural Controllers for a Pneumatic Actuator

Tiberiu Vesselenyi, Simona Dziţac, Ioan Dziţac, Mişu-Jan Manolescu

Abstract: There is a great diversity of ways to use fuzzy inference in robot control systems, either in the place where it is applied in the control scheme or in the form or type of inference algorithms used. On the other hand, artificial neural networks ability to simulate nonlinear systems is used in different researches in order to develop automated control systems of industrial processes. In these applications of neural networks, there are two important steps: system identification (development of neural process model) and development of control (definition of neural control structure). In this paper we present some modelling applications, which uses fuzzy and neural controllers, developed on a pneumatic actuator containing a force and a position sensor, which can be used for robotic grinding operations. Following the simulation one of the algorithms was tested on an experimental setup. The paper also presents the development of a NARMA-L2 neural controller for a pneumatic actuator using position feedback. The structure had been trained and validated, obtaining good results.
Keywords: fuzzy control, neural control, force-position feedback, pneumatic actuator.

1 Introduction

There is a great diversity in which fuzzy inference and neural networks can be used in robotics operation control either in the place it has in the control scheme or in the type of fuzzy or neural controller. From the studied references the conclusion can be drawn that fuzzy inference is used (among others) in trajectory generation [3], robot model design [2], instead of P.I.D. controllers [4] or in combination with these [6]. A detailed presentation of general purpose fuzzy controllers is given in [5]. In the same work, it is shown that there can be made fuzzy controllers similar to classical ones (quasi - P.I.D.). In other researches the importance of parameter adjustment is emphasized and also that fuzzy controllers can be adjusted more easily [6]. Due to the fact that a large part of fuzzy inference systems had been implemented on heuristic basis (usually the membership functions are chosen upon the educated guess of specialists) there is no guarantee of a reliable operation or stability of the system in unforeseen conditions. Due to this, experimental tests must be considered. A great number of researches in this field have as goal the development of methodologies of synthesis and analysis of fuzzy inference systems, in the field of robotics [2], [4] or in the larger field of control systems [5] (i.e. study of stability of fuzzy controllers).

Also there are works regarding the elaboration of fuzzy models of robots (used in direct and inverse kinematics [3] or in inverse dynamics [1], which can replace analytical models, and shorten the computing times. Many researches try a systematic approach of fuzzy systems design (development of a design methodology), which can eliminate the subjectivity in choosing the membership functions and rule sets, as in [2], in which a clear method is presented for a rigorous selection of fuzzy inference parameters.

In order to develop a model and test fuzzy and neural control in the design phase an adequate programming environment must be selected. For this purpose we had chosen the MATLAB programming environment, because it offers predefined functions to develop fuzzy and neural control systems. These functions are linked to extern modules like the "inference system" and the "fuzzy engine", and the SIMULINK module can also use these functions. User applications can be linked to these modules using the predefined functions.

The typical base structure of fuzzy systems develop a model which make the correspondence:

- crisp value - input membership functions - inference rules -
- output characteristics - output membership functions - crisp output value.

Also, a typical fuzzy inference system, supposes a user defined set of parameters which try to encrypt the model's variables characteristics. If instead the development of a process model is wanted for which certain experimental input-output data sets exists, the fuzzy system parameters can be automatically generated that is the system identification can be done. In this case the identification strategy can be a neural-fuzzy approach, which has at its base acquiring knowledge from the presented data set in order to generate membership function parameters. In the MATLAB environment the adjustment of these parameters can be done with a module which works similar to a neural network named ANFIS (Adaptive Neural Fuzzy Inference System). As teaching algorithm error back propagation is used and the optimization is made by a gradient method, followed by error minimization (by the quadratic sum method).

In [10], a methodology is presented for designing an adaptive fuzzy logic controller. "The neuro-fuzzy controller is first trained using data from an approximate analytical model of a cellular network then the controller is fine tuned and adapted to the unique cell dwell time and call holding time distributions of a particular cell in the network".

The ability of neural networks to simulate non-linear systems, is used in some researches [13], in order to develop industrial processes control systems. When using neural networks in controlling processes there have to be two steps: system identification (development of process neural model) and control design (development of neural control system).

In the system identification step, the neural model of the controlled process is developed and in the second step this model is used to obtain the neural net that will control the process. The training of process neural model is made "offline" (or "batch processing"), but the training and optimization of neural control must be made "online" using training data sets.

A chaos search immune algorithm is proposed in [12] by integrating the chaos optimization algorithm and the clonal selection algorithm: "first, optimization variables are expressed by chaotic variables through solution space transformation. Then, taking advantages of the ergodic and stochastic properties of chaotic variables, a chaos search is performed in the neighborhoods of high affinity antibodies to exploit local solution space, and the motion of the chaotic variables in their ergodic space is used to explore the whole solution space. Furthermore, a generalized radial basis function neuro-fuzzy controller [...] is constructed and designed automatically".

2 Simulation of pneumatic system with fuzzy controller

2.1 General considerations

The general scheme of the automated grinding system is presented in figure 1. This system is used to grind metal probes for microscope observations.

The "Command module" represents a programmable computing unit on which the control algorithm is running (fuzzy or neural controller - in this case a Pentium IV PC) and it has the possibility to transmit signals to the execution unit and to receive data from sensors (by means of a DAQ card with analogical and digital channels).

The "Execution module" contains the execution elements, which are simple and proportional electro - valves, "Force sensor" and "Position sensor" - are the sensor used to generate the feedback signals.

The pneumatic setup of the force-position feedback system (FPFS) is shown in figure 3. For the position feedback (PFS) system the force sensor is missing.

In the position feedback case the system has to move the probe, approaching the grinding surface to an approximate distance, and then to move it with smaller speed to touch the surface.

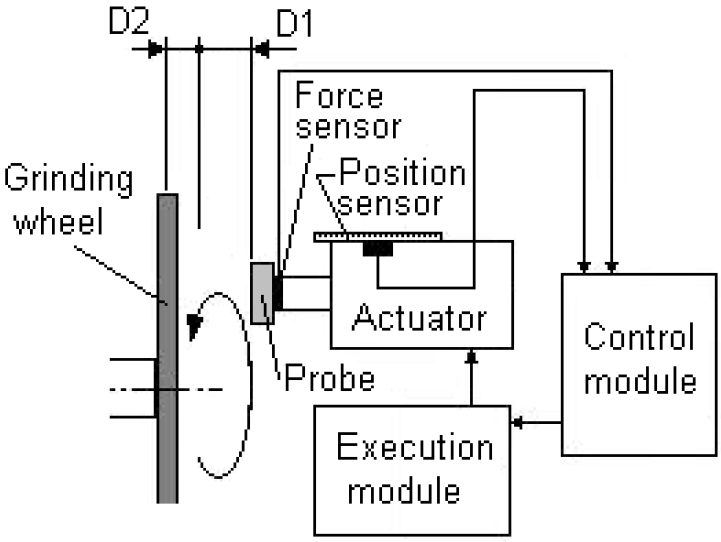


Figure 1: Scheme of automated metal probe grinding system

In the force - position feedback case the system has to fulfil the objectives represented in diagram presented in figure 2, that is to move the probe near the grinding wheel surface and then to move it with reduced speed until the force reaches a reference value and then again to maintain the probe pushed on the surface with the reference force. Above described strategies can be graphically expressed in the diagram shown in figure 3.

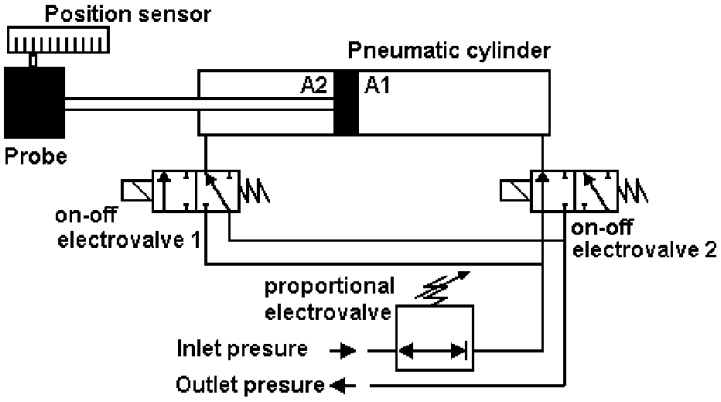


Figure 2: Scheme of pneumatic setup

For the fuzzy FPFS case (the complete case) there have to be defined:

- two input linguistic variables "Position error" and "Forces error" and one output linguistic variable named "Control value".
- position reference is given by the superior limit of the uncertainty domain D2;
- the final control will be made by the "Force" variable;
- for position there are two ranges which must be defined (Big and Small) and for the force seven ranges (Big negative, Medium negative, Small negative, Zero, Big positive, Medium positive, Small positive).

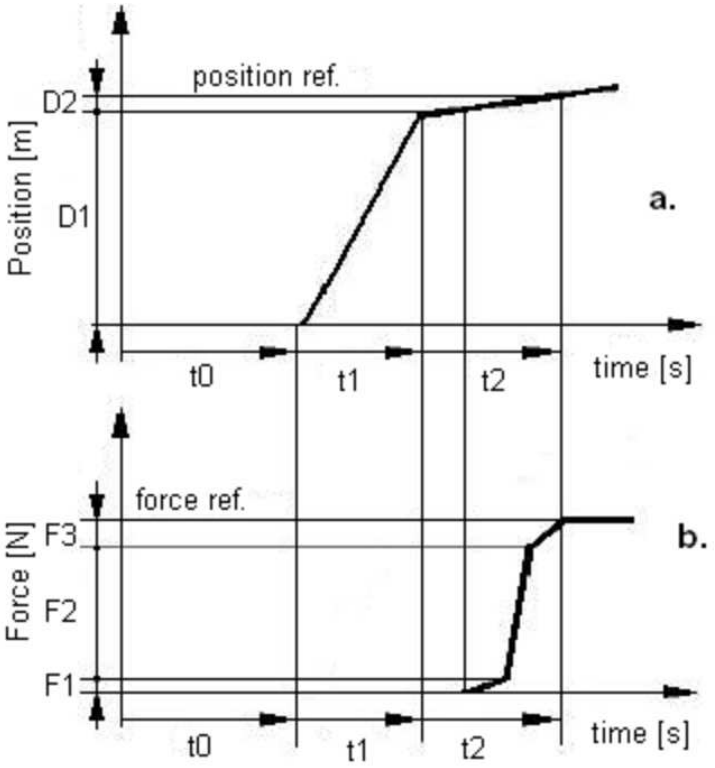


Figure 3: Functional diagrams for position (a) and force (b) control

2.2 Fuzzy position feedback system

The model for fuzzy PFS is presented in figure 4. In this case the system is used without force reaction. This model has been made only as a preliminary study.

In figure 5 the systems inference diagrams are shown:

- the gaussian input membership functions represents 5 linguistic ranges of the "Position error";
- the output triangular membership functions represent also 5 linguistic ranges of the output variable "Control value"; deffuzyfication is made by a "centroid" function.

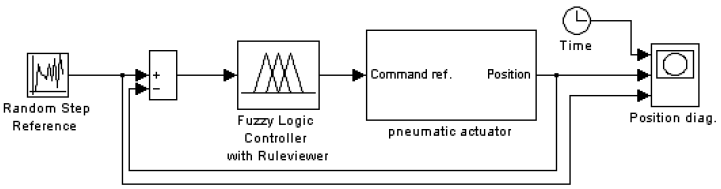


Figure 4: Fuzzy PFS model

Results of simulation is given in figure 6 for some random step reference values. We can observe how the actual controlled position is following the reference position. The delays are relatively large but considering that the pneumatic system is acting similar to a damper, this range of delay is acceptable.

2.3 Fuzzy force - position feedback system

The fuzzy FPFS is presented in figure 7.

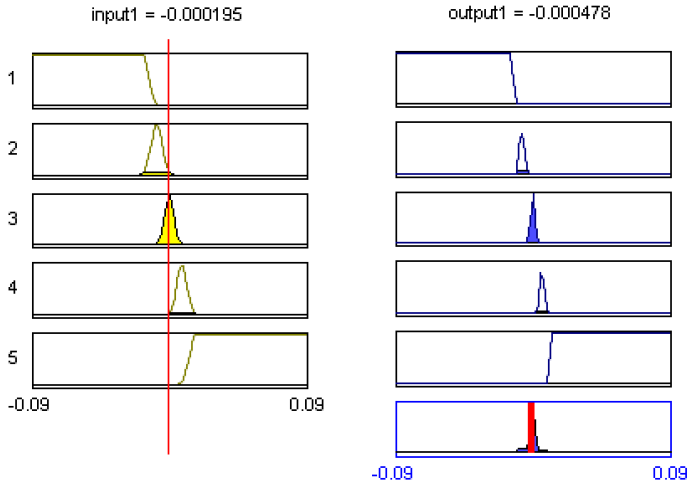


Figure 5: Inference diagram of the fuzzy PFS model

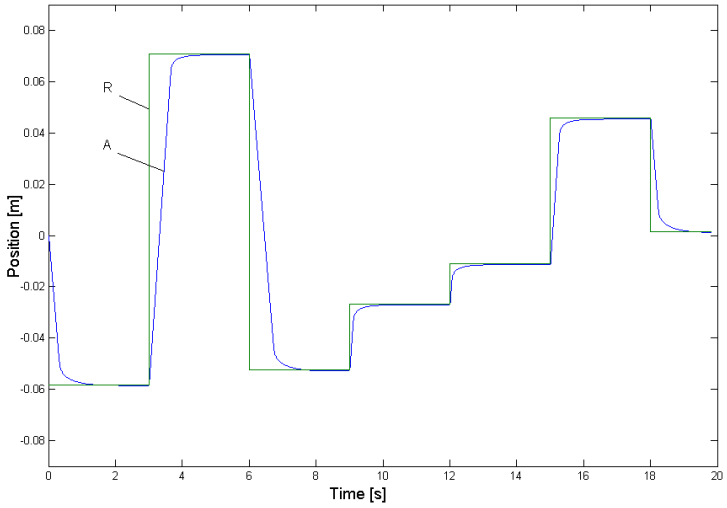


Figure 6: Reference (R) and actual (A) signals diagram resulted from system simulation (fuzzy PFS case)

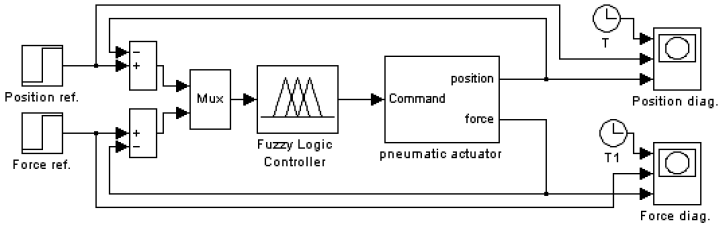


Figure 7: Fuzzy FPFS model

For both inputs (as for "Force error" as for "Position error") the membership functions are "gaussian" type and for output membership function the "triangular" type had been used. Simulation results for step type references are given in figures 8 and 9.

Analyzing the result diagram we can conclude that the control is working correctly.

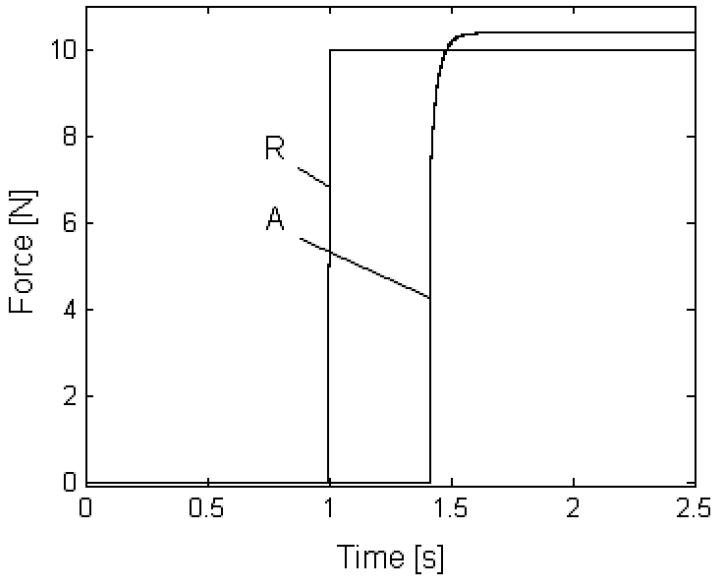


Figure 8: Reference and response force values

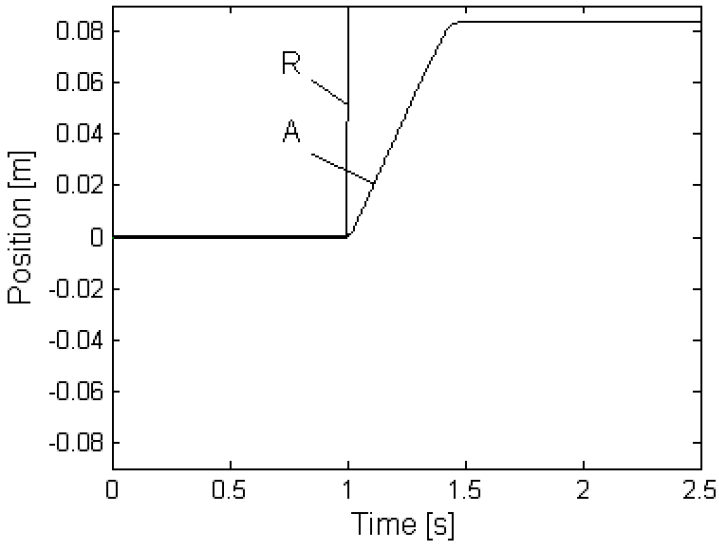


Figure 9: Reference and response position values

3 Experimental setup for fuzzy FPFS

The experimental study in order to test fuzzy FPFS operation is shown in figure 10. The base idea of this concept was the use of a PCI6023E DAQ card (from National Instruments), for which there are pre-defined acquisition functions in the MATLAB "Data Acquisition Toolbox". This fact makes possible the

use of data acquisition, fuzzy inference engine and command signal generation from the same program. Even the designed system gives birth to considerable delays in comparison with a true real time system (which would use xPCTarget modules), it is good enough for testing of controllers accuracy.

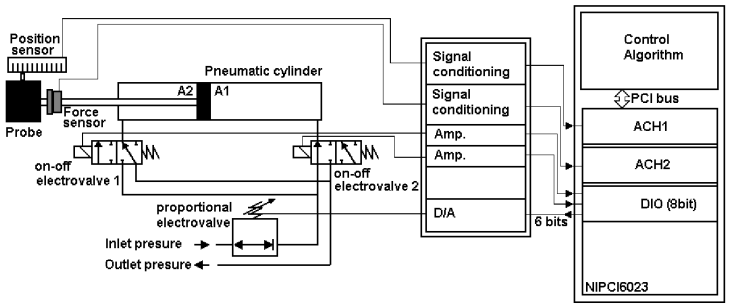


Figure 10: Experimental setup scheme

The DAQ card has 8 analogical input channels (ACH1...8) on 12 bits, from which ACH1 and ACH2. were used.

A major disadvantage of PCI6023 DAQ card is the absence of analog output, which would be used to command the pressure regulator proportional electro-valve (figure 10). This issue was solved by using 6 out of 8 digital channels of the digital I/O port (DIO in figure 10) and a D/A (digital to analog) converter. So we can obtain 64 values of pressure for the 6 bits available, which suffice for the experiments. We cannot use al the 8 cannels of the DIO port because 2 channels must be used to command the 2 on-off electro-valves used to change the piston's movement direction.

4 Neural position feedback system

In the case of neural PFS there must be two steps to complete: system identification (development of process neural model) and control design (development of neural control system) [3].

In the system identification step, the neural model of the controlled process is developed and in the second step this model is used to obtain the neural net that will control the process.

4.1 Identification step

In the identification step a convenient structure for the process model must be found and then the neural network will be trained in order to obtain the value of weights, using training data sets. A largely used standard structure, representative for nonlinear discrete systems is NARMA (Nonlinear AutoRegressive - Moving Average) [3], given by the relation:

$$y(k+d) = N[y(k), y(k-1), \ldots, y(k-n+1), u(k), u(k-1), \ldots, u(k-n+1)] \quad (1)$$

in which: $u(k)$ is the system's input and $y(k)$ is the output. In order to identify the process, the network will be trained with the non-linear function N.

4.2 Control step

If the goal of the system is to follow a reference trajectory, $y(k+d) = y_r(k+d)$, a non-linear controller will have to be developed:

$$u(k) = G[y(k), y(k-1), \ldots, y(k-n+1), y_r(k+d), u(k-1), \ldots, u(k-m+1)] \quad (2)$$

In order to generate the function G, which minimizes the quadratic mean error, a dynamic back-propagation learning algorithm should be used which is hard to implement and very slow. That is the reason why some approximate models are usually used.

Such an approximate model is given by relation:

$$\begin{aligned} y(k+d) &= f[y(k),y(k-1),\ldots,y(k-n+1),u(k-1),\ldots,u(k-m+1)] \\ &\quad +g[y(k),y(k-1),\ldots,y(k-n+1),u(k-1),\ldots,u(k-m+1)]\cdot u(k) \end{aligned} \tag{3}$$

This model is in a form in which the input u(k) is not contained in the non-linear term and if $y(k+d)=y_r(k+d)$, it can then be written that:

$$u(k)=\frac{y_r(k+d)-f[y(k),y(k-1),\ldots,y(k-n+1),u(k-1),\ldots,u(k-n+1)]}{g[y(k),y(k-1),\ldots,y(k-n+1),u(k-1),\ldots,u(k-n+1)]} \tag{4}$$

In this form it is necessary to find the input values $u(k)$, based on the output in the same step $y(k)$, which is inconvenient and is better to use the form:

$$\begin{aligned} y(k+d) &= f[y(k),y(k-1),\ldots,y(k-n+1),u(k),u(k-1),\ldots,u(k-n+1)] \\ &\quad +g[y(k),y(k-1),\ldots,y(k-n+1),u(k),\ldots,u(k-n+1)]\cdot u(k+1) \end{aligned} \tag{5}$$

for $d \geq 2$.

The structure of the process model neural network neural network is given in figure 11.

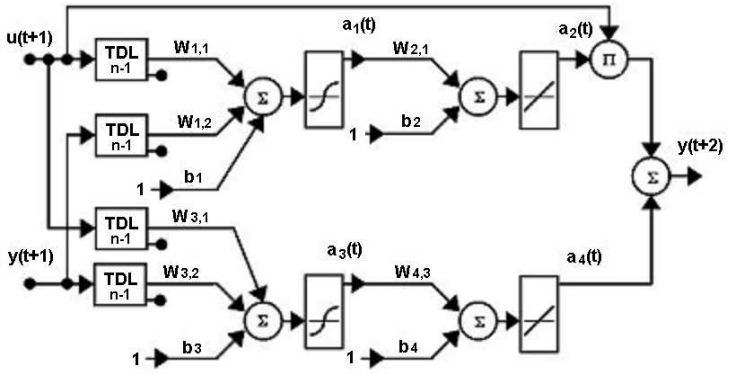


Figure 11: Process neural model based on relation (5)

The controller expression from relation 5, will be:

$$u(k+1)=\frac{y_r(k+d)-f[y(k),\ldots,y(k-n+1),u(k),\ldots,u(k-n+1)]}{g[y(k),\ldots,y(k-n+1),u(k),\ldots,u(k-n+1)]} \tag{6}$$

for $d \geq 2$.

In figure 12 is presented the control scheme in which yr is generated by the neural model ("reference model").

In this case the controller has to make only a few computations and the neural model can be trained off-line. The method can be applied in industrial robot control, but it was not tested yet with pneumatic actuators. The real advantage of using this kind of controllers for industrial robots, would be the use of parallel system for each axes of the robot.

As support of computational implementation of the simulation the SIMULINK module of MATLAB was used. SIMULINK, contains predefined blocks to generate the neural model as well a GUI for parameter settings, training and validation of the controller.

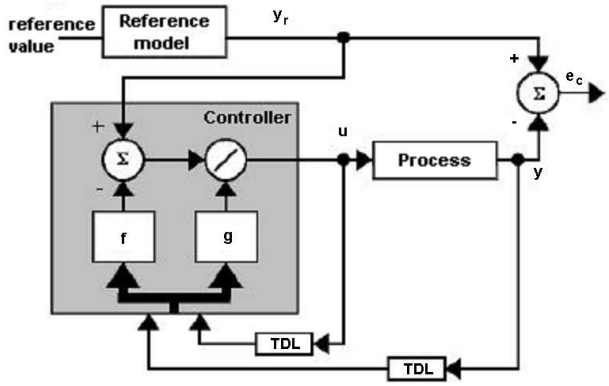


Figure 12: Control scheme designed on the relation (6)

4.3 Neural position feedback system simulation

For this case a proportional valve and two on-off electro-valves have been used. The pneumatic design scheme is the same as it was presented in figure 2, only the controller structure is different.

This model is very complex and even if the PC used for simulation (Intel Pentium IV 2GHz), has a significant computing power training cycles are very long (about hundreds of minutes). That is why we have used the simplified model of the process shown in figure 13. The simplification is made by neglecting the air compressibility terms and the elimination of air from cylinders inactive chamber.

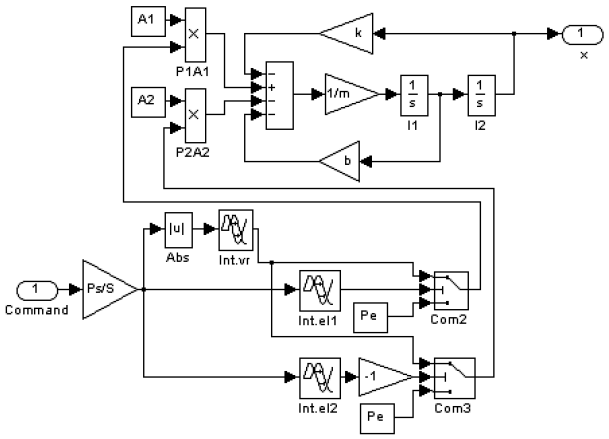


Figure 13: Simplified process scheme

The presented model is then integrated with the controller model given the final version shown in figure 14.

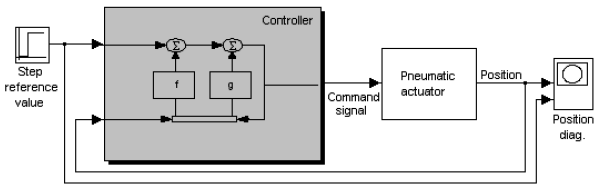


Figure 14: Process with controller scheme

After model design, the training, validation and test data sets were generated with the help of process model. On this basis the neural model of process and controller were obtained. The training error as function of training epochs is presented in figure 15.

A number of 3000 data in 300 epochs were used as training parameters. As it can be seen from the diagram the convergence of the network is good.

Simulations were carried out for different reference values for position from 0.025 to 0.06. In figure 16 a sample of obtained response is shown for position reference value of 0.042.

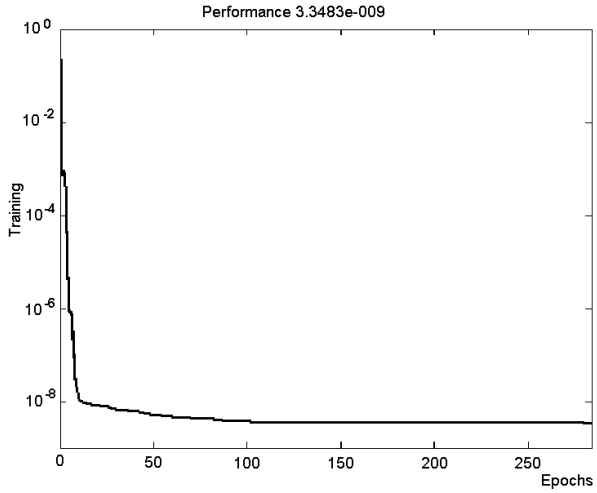


Figure 15: Training of the neural network

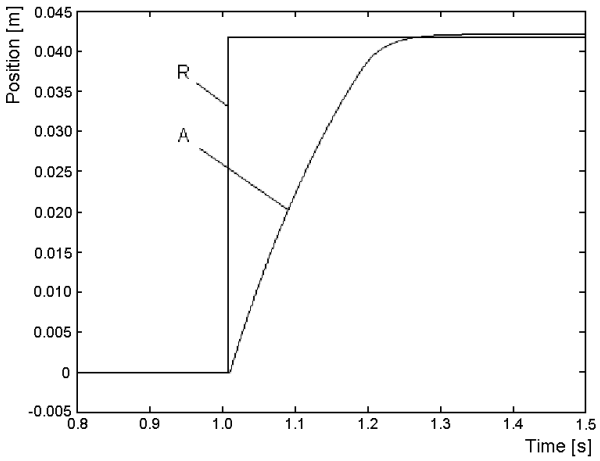


Figure 16: Reference (R) and actual (A)value of simulation results

5 Results and conclusions

The structure of the NARMA-L2 neural controller for a pneumatic actuator has been trained and validated, obtaining good results.

After the analysis of simulation results and the experimental system operation it can be said that the fuzzy controller is working in a proper manner for this application but needs further adjustments in order to increase the robustness of the control.

In actual operation there are unwanted effects as:

- position control error is constant, but the actual position remains under the reference value with about 6% for a value of reference from 6 to 9;
- the uncertainty domain in which the probe can meet the grinding wheel and the domain in which the control is acceptable is small (about 0.003 m), but in practice this is about 0.005 to 0.006 m;
- the force control error is increasing with the increase of reference, limiting the application domain in which the error is still acceptable.

Although in some domains the control is acceptable further adjustments of the control parameters are needed.

In the study of neural controller simulation it resulted that the overshoot error of the system was only of 3%. In this case further studies must be carried out to implement the neural controller for the FPFS case and to test it experimentally.

Bibliography

[1] Amann P., Perronne J.,M., Gissinger G.,L., Frank P., M., Identification of fuzzy relational models for fault detection, *Control Engineering Practice* 9, 555, 2001.

[2] Emami M.R., Goldenberg A.A., Burhan T.R., Systematic design and analysis of fuzzy-logic control and application to robotics, Part I. *Modeling, Robotics and Autonomous Systems* 33, pp. 65-88, 2000.

[3] Jang, R., *MATLAB - Fuzzy Toolbox - The MathWorks, Inc. Revision: 1.12*, Date: 2000, 15.

[4] Novakovic, B., Scap, D., Novakovic D., An analytic approach to fuzzy robot control synthesis, *Engineering Applications of Artificial Intelligence* 13, pp. 71-83, 2000.

[5] Preitl, St., Precup, E., *Introducerea în conducerea fuzzy a proceselor*, Ed. Tehnicã, Bucuresti, 1997.

[6] Reznik L., Ghanayem O., Bourmistrov A., PID plus fuzzy controller structures as a design base for industrial applications, *Engineering Applications of Artificial Intelligence* 13, pp. 419-430, 2000.

[7] Vesselenyi T., *Automated flexible cell for microstructure recognition*, PhD Thesis, Universitatea "Politehnica" Timisoara, 2005.

[8] M.M., Chen, J.A., Fairwather, S.A., Green, EDUMECH. *Mechatronic Instructional Systems. Case Study: Pneumatics Systems, Production of Shandor Motion Systems, Inc.*, 1999.

[9] Harbick K., Sukhatme S., *Speed Control of a Pneumatic Monopod using a Neural Network*, www.harbick-ann.com, 2002.

[10] Raad R., Raad I., *Neuro-Fuzzy Admission Control in Cellular Networks. Communication systems*, (10th IEEE Singapore International Conference on Communication systems.), pp. 1-7, 2006.

[11] Wenmei H., Yong Y., Yali T., *Adaptive neuron control based on predictive model in pneumatic servo system*, 2002.

[12] Zuo X.Q., Fan Y.S., A chaos search immune algorithm with its application to neuro-fuzzy controller design. *Chaos, Solitons & Fractals*, Vol. 30, Issue 1, pp. 94-109, 2006.

[13] Zhang J., Knoll A., Schmidt R., A neuro-fuzzy control model for fine-positioning of manipulators, *Robotics and Autonomous Systems*, 32, pp. 101-113, 2000.

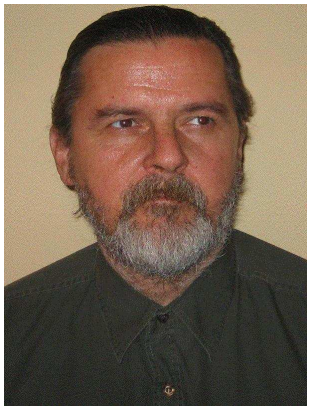
Tiberiu Vesselenyi
University of Oradea
Universităţii St. 1, 410087, Oradea, Romania
tvesselenyi@yahoo.co.uk

Simona Dziţac
University of Oradea
Universităţii St. 1, 410087, Oradea, Romania
sdzitac@rdslink.ro

Ioan Dziţac
Department of Economic Informatics
Agora University of Oradea
Piaţa Tineretului 8, Oradea 410526, Romania
idzitac@univagora.ro

Mişu-Jan Manolescu
Agora University
Piaţa Tineretului 8, 410526 Oradea, Romania
rectorat@univagora.ro

Received: February, 14, 2007



Tiberiu Vesselenyi was born in Oradea, Romania in 1957, he finished the University "Politehnica" from Timişoara in 1983. From 1983 to 1991 he worked at a machine building company in Oradea as designer and CNC programmer. From 1991 till 1994 he was a research engineer at the "Geothermal Energy Research Center" in Oradea and from 1994 till today is assoc. prof. at the University Of Oradea, where he teaches robot and CNC programming. He had earned a PhD in robotics at the University "Politehnica" at Timişoara. He had published over 150 papers in national and international conferences and journals, and is author or coo - author of 4 books.

Simona Dziţac received B.Sc. (2000) and M. Sc. (2001) in Mathematics-Physics, B.Sc. (2005) and M. Sc. (2007) in Energy Engineering from University of Oradea and B.Sc. in Economic Informatics (2007) from University of Craiova, Romania. At this moment, she is PhD student in Energy Engineering field and researcher at University of Oradea. Her current research interests include Reliability, Applied Mathematics and Computer Science in Engineering fields. She published 4 books and 38 scientific papers in journals and conferences proceedings.

Ioan Dziţac received M. Sc. in Mathematics (1977) and Ph. D in Information Sc. (2002) from "Babes-Bolyai" University of Cluj-Napoca. At this moment, he is associate professor and head of Economic Informatics Department at AGORA University, Oradea, Romania. His current research interests include different aspects of Parallel and Distributed Computing, Applied Mathematics and Economic Informatics. He has edited 4 conference proceedings, published 12 books and 47 scientific papers in journals and conferences proceedings. He was member of the Program Committee of 18 international conferences.

Mişu-Jan Manolescu received M. Sc in Electro-mechanics (1984) from the University of Craiova, Ph. D in Microwave (1994) from the University of Oradea, and Ph. D in Human Resources Management (2000) from the University of Craiova. Now, he is professor and president of AGORA University, Oradea, Romania. His current research interests include different aspects of Knowledge Management, and Knowledge Engineering. He has edited 4 conference proceedings, published 9 books and 63 scientific papers in journals and conferences proceedings. He has been member of Program Committee of the 5 international conferences.

International Journal of Computers, Communications & Control
Vol. II (2007), No. 4, pp. 388-402

A Toolbox for Input-Output System Inversion

Antonio Visioli, Aurelio Piazzi

Abstract: In this paper a Matlab-based toolbox for the input-output system inversion of linear systems is presented. Different methods, based either on analytical or numerical approaches, are implemented. The toolbox can be exploited in the design of a feedforward action for control systems in different contexts in order to improve performances in the set-point regulation. The use of a pre-actuation and a post-actuation time can be easily analyzed as well as the role played by the choice of the desired output function.
Keywords: CACSD, input-output inversion, feedforward, set-point regulation, optimization.

1 Introduction

It is well-known that a (properly designed) feedback controller provides robustness to a control system with respect to parameter variations and allows to compensate for external disturbances. On the other side, a high performance in the set-point following task can be achieved by adopting a suitable feedforward action. Indeed, the proper design of a control system consists of suitably combining feedback and feedforward control. Different techniques have been developed for the synthesis of a feedforward controller for a linear system (see e.g. [1, 2, 3]).
When the set-point regulation is of concern, a (noncausal) system inversion approach has been proven to be effective in this context [4]-[13]. Basically, the approach consists of selecting a desired output function in order to achieve a transition from a current output value y_0 to a new one y_1 and then to determine the corresponding input function by applying a stable inversion procedure. Then, the calculated input function is adopted as a reference command input to the (closed-loop) system, instead of the typical step signal.
Actually, while many software packages are available for the synthesis of feedback controllers (for example, via root locus techniques or Bode plots), they are not available for the synthesis of a system inversion based feedforward action. Indeed, the presence of a Computer Aided Control Systems Design tool makes the applicability of these (somewhat complex) techniques much easier and it can be exploited to understand deeply the role of the command function in the context of set-point regulation.
In this paper a Matlab-based toolbox for the input-output system inversion of a linear system is presented. Different techniques in this context are considered, related both to a noncausal and a causal approach. The toolbox allows to evaluate the role of the use of a pre-actuation and a post-actuation time as well as the role of the choice of the desired output function. It can be adopted as a useful tool in different fields, such as robust control, process control and control of mechatronic systems.
The paper is organized as follows. In Section 2 the input-output system inversion approach is briefly reviewed and the different methodologies implemented in the toolbox are presented. The functions implemented in the toolbox are described in Section 3 and application examples are shown in Section 4. Conclusions are drawn in Section 5.
Notation. C^i denotes the set of scalar real functions that are continuous till the ith derivative and BC^i denotes the subset of C^i of the scalar real functions that are bounded. The ith order differential operator is D^i.

2 Input-output system inversion

Consider a general asimptotically stable nonminimum-phase linear system Σ described by the following rational transfer function (note that this might represent the transfer function of a feedback control system):

$$H(s) = K_1 \frac{b(s)}{a(s)} = K_1 \frac{s^m + b_{m-1}s^{m-1} + \cdots + b_0}{s^n + a_{n-1}s^{n-1} + \cdots + a_0}, \quad K_1 \neq 0 \tag{1}$$

where it is assumed that polynomials $a(s)$ and $b(s)$ are coprime (no pole-zero cancellations occur) and that Σ has not purely imaginary zeros.
The input and output of Σ are $u \in \mathbb{R}$ and $y \in \mathbb{R}$ respectively and the relative order (or relative degree) of Σ is $\rho := n - m$. The set of all cause/effect pairs associated with Σ is denoted by

$$\mathcal{B} := \{(u(\cdot), y(\cdot)) \in P_c \times P_c : D^n y + a_{n-1}D^{n-1}y + \cdots + a_0 y = K_1(D^m u + b_{m-1}D^{m-1}u + \cdots + b_0 u)\} \tag{2}$$

where P_c denotes the set of piecewise continuous functions defined over $(-\infty, +\infty)$, i.e. the real field $\mathbb{R}$. In the framework of the behavioral approach, $\mathcal{B}$ is the behavior set of Σ that can be rigorously introduced by means of the so-called *weak solutions* of the differential equation associated to Σ [14].
The following proposition [14] is useful in the development of the subsequent analysis.

Proposition 1. *Consider any pair $(u(\cdot), y(\cdot)) \in \mathcal{B}$. Then, $u(\cdot) \in C^l(\mathbb{R})$ if and only if $y(\cdot) \in C^{\rho+l}(\mathbb{R})$ with l being a nonnegative integer.*

The considered regulation problem consists of obtaining an output transition from a previous value y_0 to a new value y_1. Without loss of generality, in the following we will consider $y_0 = 0$. Define $y_d(\cdot) \in BC^k$ with $y_d(t) = 0$ for $t < 0$ as the desired output function to obtain the transition. From a practical point of view, a transition time τ has to be defined, i.e. the desired output function is defined as

$$y_d(t) := \begin{cases} 0 & \text{for} \quad t < 0 \\ y_{01}(t) & \text{for} \quad 0 \leq t \leq \tau \\ y_1 & \text{for} \quad t > \tau. \end{cases} \tag{3}$$

Then, the following stable input-output inversion (SIOI) problem can be formulated.
SIOI problem. Determine an input function $u_d(\cdot) \in BC^{k-\rho}$ such that

$$(u_d(\cdot), y_d(\cdot)) \in \mathcal{B}. \tag{4}$$

The general solution to the SIOI problem can be derived as follows [15]. First, express the inverse of the transfer function (1) as

$$H^{-1}(s) = \frac{1}{K_1}\frac{a(s)}{b(s)} = \xi_\rho s^\rho + \xi_{\rho-1}s^{\rho-1} + \cdots + \xi_0 + H_0(s) \tag{5}$$

where $H_0(s)$ is a strictly proper rational transfer function representing the zero dynamics of Σ. By using the fraction expansion, $H_0(s)$ can be decomposed as

$$H_0(s) = H_0^-(s) + H_0^+(s) = \frac{d(s)}{b^-(s)} + \frac{e(s)}{b^+(s)} \tag{6}$$

where $b^-(s)$ and $b^+(s)$ are the monic polynomials containing the roots of $b(s)$ with negative and positive real parts, respectively. Define $\eta_0^-(t)$ and $\eta_0^+(t)$ as the analytic extensions of $\mathcal{L}^{-1}[H_0^-(s)]$ and $\mathcal{L}^{-1}[H_0^+(s)]$ over the space of the Bohl functions for which $\eta_0^-(t)1(t) = \mathcal{L}^{-1}[H_0^-(s)]$ and $\eta_0^+(t)1(t) =$

$\mathcal{L}^{-1}[H_0^+(s)]$ respectively.
Then, the solution of the SIOI problem is derived as:

$$u_d(t) = \xi_\rho D^\rho y_d(t) + \cdots + \xi_1 D y_d(t) + \xi_0 y_d(t) + \int_0^t \eta^-(t-v) y_d(v) dv - \int_t^{+\infty} \eta^+(t-v) y_d(v) dv. \quad (7)$$

It is worth noting that, in general, $u_d(t)$ is defined over the time interval $(-\infty, +\infty)$ and therefore, in order to practically use it, it is necessary to truncate it. Thus, the input function exhibits a pre-actuation (associated with the unstable zeros) and a post-actuation (associated with the stable zeros) time intervals (see for example [16]), denoted as t_p and t_f respectively. They can be calculated with arbitrary precision by selecting two arbitrary small parameters ε_0 and ε_1 and by subsequently determining

$$t_0 := \max\{t' \in \mathbb{R} : |u_d(t)| \leq \varepsilon_0 \ \ \forall t \in (-\infty, t']\} \quad (8)$$

and

$$t_1 := \min\left\{t' \in \mathbb{R} : \left|u_d(t) - \frac{1}{H(0)}\right| \leq \varepsilon_1 \ \ \forall t \in [t', \infty)\right\}. \quad (9)$$

Then, it has to be fixed

$$t_p = \min\{0, t_0\} \qquad t_f = \max\{\tau, t_1\}. \quad (10)$$

Hence, the actual input function to be applied to the system is given by

$$\tilde{u}_d(t) := \begin{cases} 0 & \text{for} \quad t < t_p \\ u_d(t) & \text{for} \quad t_p \leq t \leq t_f \\ \frac{1}{y_1} & \text{for} \quad t > t_f. \end{cases} \quad (11)$$

Alternatively, the pre-actuation and post-actuation time intervals can be calculated as [4]

$$t_p = -\frac{10}{D_{rhp}} \qquad t_f = \frac{10}{D_{lhp}} \quad (12)$$

where D_{rhp} and D_{lhp} are the minimum distance of the right and left half plane zeros respectively from the imaginary axis of the complex plane.
It is worth noting that the pre-actuation time is zero when there are no unstable zeros and the post-actuation time is zero when there are no stable zeros.
In general, the integrals in expression (7) can not be solved analytically and therefore a numerical solution has to be determined (in this context the Cavalieri-Simpson's rule can be conveniently exploited to found an accurate solution with a small computational time [15]). A notable exception occurs when the selected desired output function is a polynomial function [17], i.e.

$$y_d(t;\tau) = \begin{cases} 0 & \text{for} \quad t < 0 \\ y_1 \frac{(2k+1)!}{k!\tau^{2k+1}} \sum_{i=0}^{k} \frac{(-1)^{k-i}}{i!(k-i)!(2k-i+1)} \tau^i t^{2k-i+1} & \text{for} \quad 0 \leq t \leq \tau \\ y_1 & \text{for} \quad t > \tau \end{cases} \quad (13)$$

Note that function $y_d(t;\tau)$, parameterized by the transition time τ is a C^k-function over $(-\infty, +\infty)$ and is strictly increasing in the interval $[0, \tau]$ so that neither overshooting nor undershooting appear in this output planning for set-point regulation. In this case an analytical solution can be found [18] and this fact can be exploited in speeding up the computational time and most of all in avoiding numerical problems.
A very interesting application of the analytical stable inversion procedure is for the improvement of the set-point following performance of Proportional-Integral-Derivative (PID) controllers. Specifically, if a PID controller is employed for a first-order plus dead-time (FOPDT) process, described by the following transfer function,

$$P(s) = \frac{K}{Ts+1} e^{-Ls}, \quad (14)$$

or for an integrator plus dead-time (IPDT) process, described by the transfer function

$$P(s) = \frac{K}{s} e^{-Ls}, \tag{15}$$

then a closed-form solution of the stable input-output inversion applied to the closed-loop system can be exploited (a rational closed-loop transfer function is obtained by adopting a Padè approximation) [13]. Indeed, the actual command signal to be applied for a given process and a given PID controller is determined by substituting the actual value of the process and PID parameters into the resulting closed-form expression. In this context the PID transfer function is expressed as

$$C(s) = K_p \left(1 + \frac{1}{T_i s} + T_d s\right) \frac{1}{T_f s + 1}, \tag{16}$$

where K_p is the proportional gain, T_i is the integral time constant, T_d is the derivative time constant and T_f is the time constant of the filter that is adopted to render the system proper.
A polynomial output function can be also usefully exploited in determining a causal input-output inversion despite the presence of unstable zeros [9]. In particular, the order of the polynomial function is selected in order to satisfy boundary conditions so that $y_d(\cdot) \in BC^k$ with $k \geq \rho$ and at the same time to have a number of free coefficients equal to the number of the unstable zeros of the system Σ. Then, the free parameters are determined in order to annihilate the unstable modes in the input function determined by the inversion procedure. In this way there is no need of a pre-actuation time interval and the resulting inversion is causal. It can be therefore employed when a preview time is not available in a given application. However, this is paid by the possible presence of undershoots and overshoots in the resulting output function. Note that the approach can be easily extended in order to avoid also the presence of a post-actuation time interval.

3 Toolbox description

The designed Matlab-based toolbox implements the methods described in Section 2. It requires the Control System Toolbox and the Symbolic Math Toolbox of Matlab.
The following main functions are available.

```
[time,command,preaction,postaction]=numdyninvcs(sys,yd,tau,st)
```

This function determines the input command function of a system that causes a desired output function by means of an input-output inversion scheme based on the use of the Cavalieri-Simpson's rule for the determination of the integrals in (7) [15]. In particular, the meaning of the parameters is the following one:

- `sys` is the transfer function of the system expressed in symbolic form (with `s` as a symbolic variable);
- `yd` is the desired output function (for $t \in [0, \tau]$) expressed as an array of numerical values from 0 to τ corresponding to the time instant equally spaced by the sampling time;
- `tau` is the transition time;
- `st` is the sampling time;
- `time` is the output time vector; it starts from the preaction time t_p but for convenience the zero time is shifted to t_p;

- `command` is the determined input function expressed as a numerical array corresponding to the time array `time`;
- `preaction` is the pre-actuation time calculated with formula (12);
- `postaction` is the post-actuation time calculated with formula (12).

`[time,command,preaction,postaction]=outdyninvcs(sys,yd,tau,st)`
This function operates basically as `numdyninvcs` with the difference that the desired output function `yd` is expressed as a symbolic expression with symbolic variable `t`.

`[time,command,preaction,postaction]=numdyninv(sys,yd,tau,st)`
This function is very similar to `numdyninvcs` but it performs the numerical integration by applying a rectangular rule. In order to obtain an accurate result, a small value of the sampling time has to be selected. This might result in a high computational time.

`[time,command,preaction,postaction]=outdyninv(sys,yd,tau,st)`
This function is very similar to `outdyninvcs` but it performs the numerical integration by applying a rectangular rule. Also in this case, in order to obtain an accurate result, a small value of the sampling time has to be selected. This might result in a high computational time.

`[time,command,preaction,¯postaction]=dyninv(sys,y1,tau,threshold0, threshold1,st)`
This function solves the input-output inversion problem when the desired output function is a polynomial function (13). The order of the polynomial is automatically selected, according to Proposition 1, in order to obtain a continuous input function, i.e. such as $u_d(\cdot) \in BC^0$. Since the input function is determined analytically, the pre-actuation and post-actuation time intervals are conveniently determined by adopting formulae (10). The function parameters that are different from those that have been already described have the following meaning:

- `y1` is the desired new output steady-state value (it is assumed, without loss of generality, that the current input and output steady-state values are zero);
- `threshold0` is the parameter ε_0 in formula (8), which is adopted to calculate the pre-actuation time;
- `threshold1` is the parameter ε_1 in formula (9), which is adopted to calculate the post-actuation time.

```
[time,command,taunum,preaction,postaction]=
              optdyninv(sys,y1,constraints,threshold0,threshold1,tc)
```

This function solves the minimum-time inversion problem that consists of finding the minimum output transition time subject to constraints posed on the input function and its derivatives until an arbitrary order l. Formally, the optimisation problem is posed as follows [18]:

$$\min_{\tau \in \mathbb{R}^+} \tau \tag{17}$$

such that, $i = 0, 1, \ldots, l$,

$$|D^i u_d(t;\tau)| \leq u_M^{(i)} \quad \forall t \in (-\infty, +\infty) \tag{18}$$

where the positive values $u_M^{(i)}$, $i = 0, 1, \ldots, l$, are given bounds of the problem. Note that the problem admits a solution if $u_M^{(0)} > 1/|H(0)|$ and $u_M^{(i)} > 0$, $i = 1, \ldots, l$. The optimisation problem is solved by applying a simple bisection algorithm in conjunction with a gridding of the time axis [17]. With respect to the function `dyninv` there are the following different parameters:

- `constraints` is the array (of $l+1$ elements) of the constraints for the input derivatives until the lth order; note that the user-chosen dimension of the array automatically determines the order of the constrained derivatives and therefore the order of the output polynomial function (which is determined as $l-1+\rho$ so that $u_d(\cdot) \in BC^{l-1}$);

- `taunum` is the resulting optimal transition time.

It is worth noting that, if a rigorous determination of the transition time is sought, the posed optimisation problem should be approached with the tools of global optimisation. In this context the presented input-output inversion toolbox can be easily integrated with the `b4m` toolbox that allows to handle interval arithmetic, which is a well-known effective tool for global optimisation [19].

`[time,command,out,postaction]=causaldyninv(sys,y1,tau,st,hbc,pa)`
This function implements the causal approach proposed in [9]. The resulting pre-actuation time is always zero despite the possible presence of unstable zeros. In particular, the function deals with the following parameters:

- `hbc` is the order h of the boundary conditions to be satisfied for the polynomial output function at time $t = 0$ and $t = \tau$, so that $y_d(\cdot) \in BC^{2h+1}$. Note that it has to be $h \geq \rho$ in order to ensure that the input function is at least continuous, i.e. $u_d(\cdot) \in BC^0$;

- `pa` is a string that, if set to `'nopostaction'`, avoid also the use of a post-actuation time even if the system has stable zeros. In other words, in this case the system attains an equilibrium point at $t = \tau$. If the parameter is not adopted or if it is set to another value, then a post-actuation time is present and it is determined by means of formula (12).

`[time,command,preaction,postaction]=piddyninvFOPDT(K,T,L,Kp,Ti,Td,Tf,tau,st)`
This function determines the input command function to unitary feedback closed-loop system in which a process described by a FOPDT transfer function is controlled by a PID controller. The following parameters are adopted:

- `K`, `T`, `L` are the process gain, time constant and dead time respectively (see (14));

- `Kp`, `Ti`, `Td`, `Tf` are the PID parameters (see (16), where the meaning of the different parameters is obvious).

`[time,command,preaction,postaction]=piddyninvIPDT(K,T,L,Kp,Ti,Td,Tf,tau,st)`
This function determines the input command function to unitary feedback closed-loop system in which a process described by a IPDT transfer function is controlled by a PID controller. The meaning of the parameters can be straightforwardly derived by considering those of the previous function together with expression (15).

4 Application examples

In order to better understand the usage of the input-output inversion toolbox, a few examples are given. Consider the system

$$H(s) = 4\frac{(1-s)(s+1)}{(s+2)(s^2+2s+2)}. \tag{19}$$

To insert the system in symbolic form in the Matlab workspace, the following two commands can be applied:

```
syms s
H=4*(1-s)*(s+1)/(s+2)/(s^2+2*s+2)
```

Then, suppose that an output transition from 0 to $y_1 = 1$ is required to be performed in $\tau = 3$ by means of the following output function:

$$\begin{aligned} y(t) \;=\; & \frac{4796646617206209}{562949953421312}t^{\frac{63}{25}} - \frac{4658008624178539}{562949953421312}t^{\frac{127}{50}} + \\ & \frac{6313836048447483}{5316911983139663491615228241121378304}t^{39}. \end{aligned} \tag{20}$$

The output function can be inserted in the Matlab workspace (denote the variable as `yt`) either in symbolic form or as an array whose elements are the values of $y(t)$ for $t = 0, T_s, 2T_s, \ldots, \tau$, where T_s is the sampling time (in this case it has been selected $T_s = 0.001$). In the first case the command to be adopted is:

```
[time,command,preaction,postaction]=outdyninvcs(H,yt,3,0.001)
```

while in the second case it has to be

```
[time,command,preaction,postaction]=numdyninvcs(H,yt,3,0.001)
```

In both cases the pre-actuation time results to be $t_p = -10$ s and the postaction time is $t_f = 10$ s (according to expressions (12)). The resulting input function and the output function obtained by applying it to the actual system are reported in Figure 1 (note that the result is the same in both cases).

Select now a polynomial output function (13) to perform again an output transition from 0 to $y_1 = 1$ and select the parameters $\varepsilon_0 = \varepsilon_1 = 10^{-3}$. In this case the Matlab command to be adopted is

```
[time,command,preaction,postaction]=dyninv(H,1,3,0.001,0.001,0.001)
```

The resulting pre-actuation and post-actuation times `preaction` and `postaction` (determined by means of formula (10)) are respectively $t_p = -6.256$ s and $t_f = 3.547$ s. The determined input and the corresponding system output are plotted in Figure 2. Note that the resulting output function is a cubic polynomial, i.e.

$$y_d(t;\tau) = y_1\left(-\frac{2}{\tau^3}t^3 + \frac{3}{\tau^2}t^2\right) \qquad t \in [0,\tau]$$

as it is $k = \rho = 1$ in formula (13).

Consider now the minimum time problem (17)-(18) and set the limits on the input derivatives as $u_M^{(0)} = 2$, $u_M^{(1)} = 10$ and $u_M^{(2)} = 20$. This means that the following Matlab command has to be executed:

```
limits=[2 10 20]
```

Then, the following function has to be employed (note that the sampling time is 0.001 s as before and again it is $\varepsilon_0 = \varepsilon_1 = 10^{-3}$):

```
[time,command,taunum,pre,post]=optdyninv(H,1,limits,0.001,0.001,0.001)
```

The resulting optimal transition time `taunum` is equal to 0.875 s, while the pre-actuation and post-actuation times are $t_p = -7.111$ s and $t_f = 4.402$ s. The determined command input and the corresponding system output are shown in Figure 3, while the first and second derivatives of the command input are plotted in Figure 4. It can be deduced that the active constraint is the one related to the second time derivative of the input.

If the causal approach is desired, i.e. the pre-actuation time is to be avoided, then the function

`causaldyninv` has to be adopted. In particular, we select the order of the boundary conditions as $h = \rho = 1$ and the desired transition time as $\tau = 3$. Then, if a post-actuation time interval is allowed, the Matlab command to be adopted is:

```
[time,command,out,postaction]=causaldyninv(H,1,3,0.001,1)
```

In this case the resulting command input and the corresponding output are plotted in Figure 5. Note that the post-actuation time interval is $t_f = 10$ s. Conversely, if a post-actuation is not allowed, the Matlab function to be employed is

```
[time,command,out,postaction]=causaldyninv(H,1,3,0.001,1,'nopostaction')
```

The resulting command input and the corresponding output function are shown in Figure 6. It can be seen that in both cases the output function is not monotonic. Indeed, the possible overshoots and undershoots are more and more significant when the selected transition time decreases.

In order to verify the effectiveness of the inversion approach for PID control, consider first the system

$$P(s) = \frac{2}{10s+1}e^{-5s}, \tag{21}$$

controlled by a PID controller (see (16)) with $K_p = 1.2$, $T_i = 10$, $T_d = 2.5$, $T_f = 0.01$. The transition time is fixed to 15 s and the sampling time to 0.01 s. Then, the following Matlab command is adopted:

```
[time,command,pre,post]=piddyninvFOPDT(2,10,5,1.2,10,2.5,0.01,15,0.01)
```

The resulting command input (applied to the closed-loop system) and output functions are plotted in Figure 7. The pre-actuation and post-actuation time intervals are $t_p = -16.67$ s and $t_f = 50.01$ s. Note that the output function is slightly different from the desired one because of the Padè approximation.

Then, a IPDT transfer function is considered:

$$P(s) = \frac{0.1}{s}e^{-5s}. \tag{22}$$

In this case the tuning of the PID parameters is $K_p = 0.12$, $T_i = 10$, $T_d = 2.5$, $T_f = 0.01$. The transition time and the sampling time are the same as before, so that the following function is employed:

```
[time,command,pre,post]=piddyninvIPDT(0.1,10,5,0.12,10,2.5,0.01,15,0.01)
```

Figure 8 reports the determined closed-loop command input and the resulting system output. Also in this case the pre-actuation and post-actuation time intervals are $t_p = -16.67$ s and $t_f = 50.01$ s.

5 Conclusions

A Matlab-based toolbox for the input-output inversion of linear scalar systems has been described in this paper. The toolbox is very useful to understand and to verify the effectiveness of a feedforward action in a control scheme and, in general, of a combined feedback/feedforward synthesis. In this context, the role played by the planned output function can be easily evaluated as well as the influence of the noncausal approach with respect to a causal one. Indeed, all the aspects related to the use of an input-output inversion-based control design can be thoroughly investigated and analyzed.

The toolbox can be exploited in different fields such as motion control, robust control, and process control.

It can be downloaded from the website http://www.ing.unibs.it/∼visioli/ioitoolbox.htm

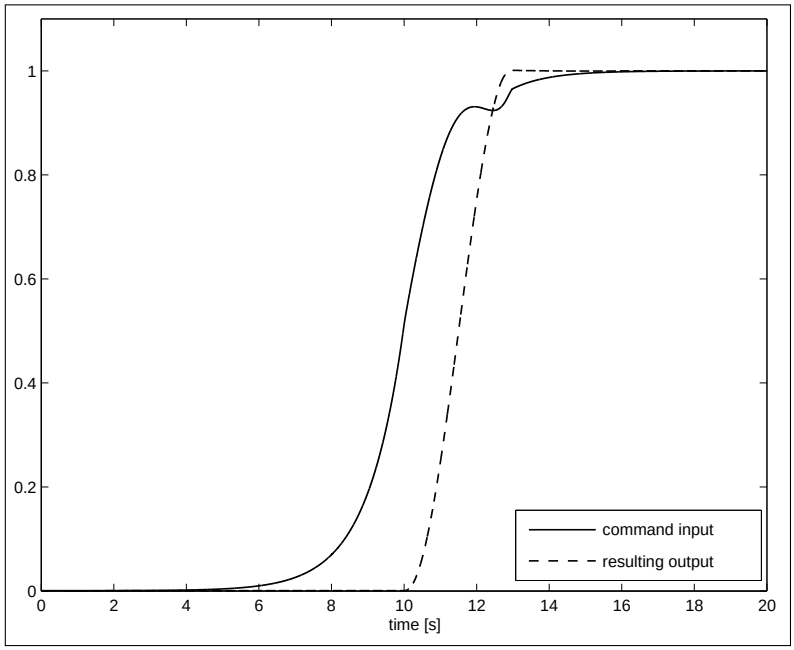


Figure 1: Command input and resulting system output with system (19) and desired output function (20)

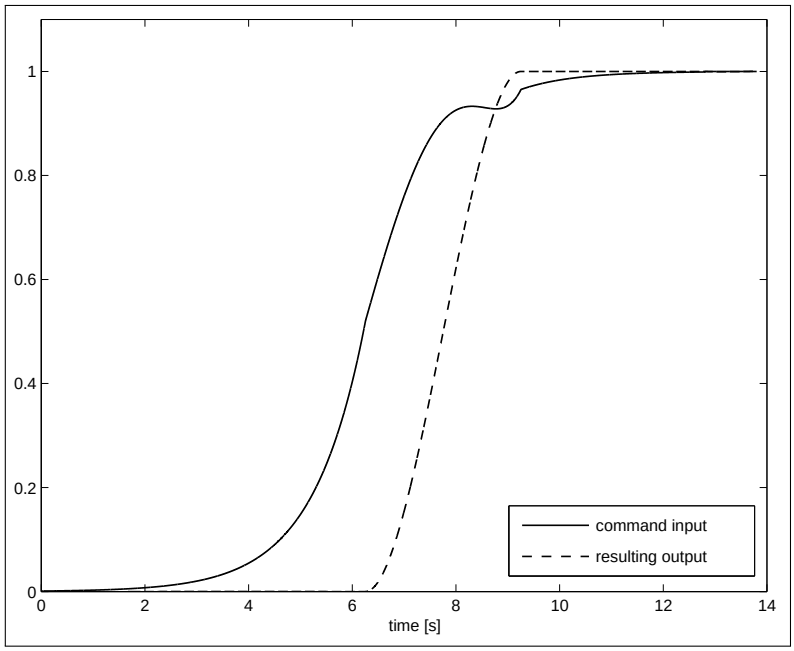


Figure 2: Command input and resulting system output with system (19) and a polynomial desired output function (13)

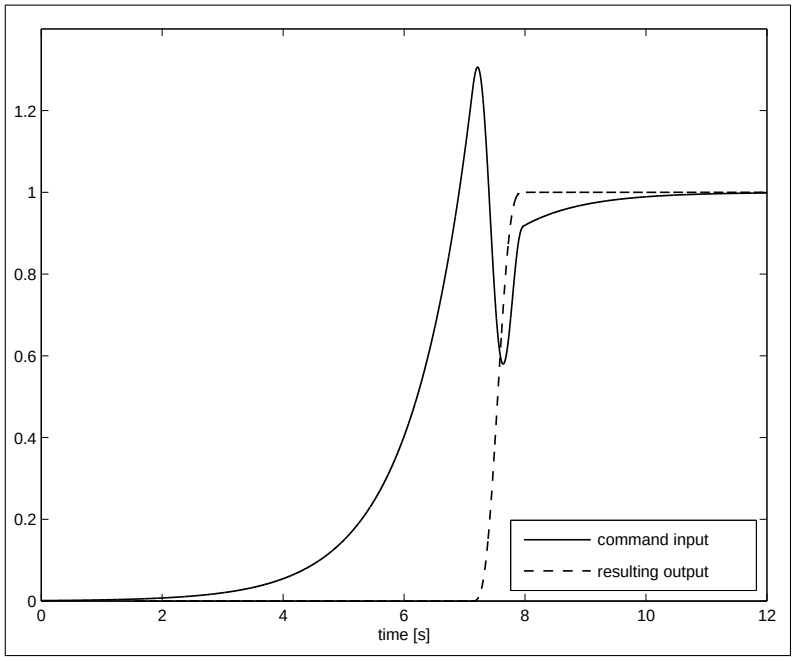


Figure 3: Optimal command input and resulting minimum-time system output with system (19) and a polynomial desired output function (13)

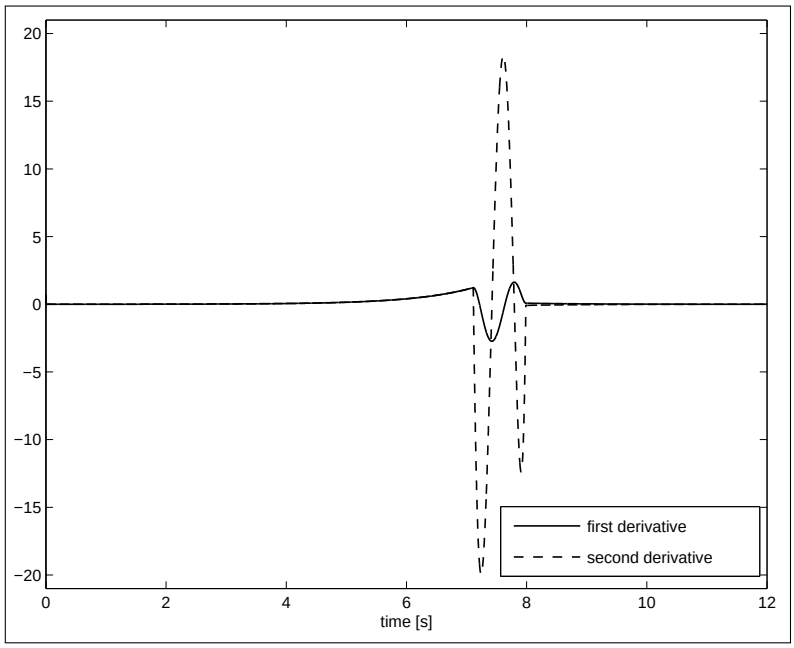


Figure 4: First and second derivative of the optimal command input with system (19) and a polynomial desired output function (13)

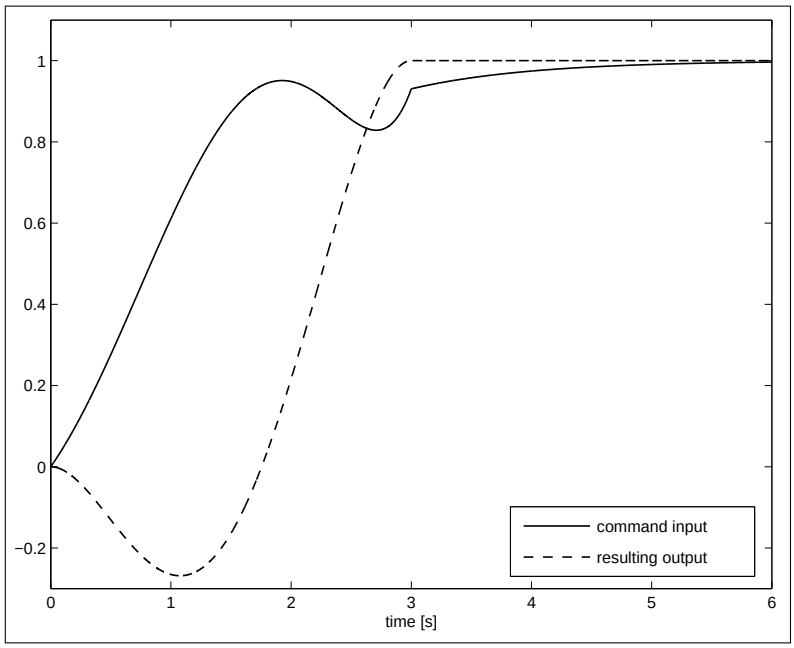


Figure 5: Command input and resulting system output with system (19) and a causal approach with post-actuation

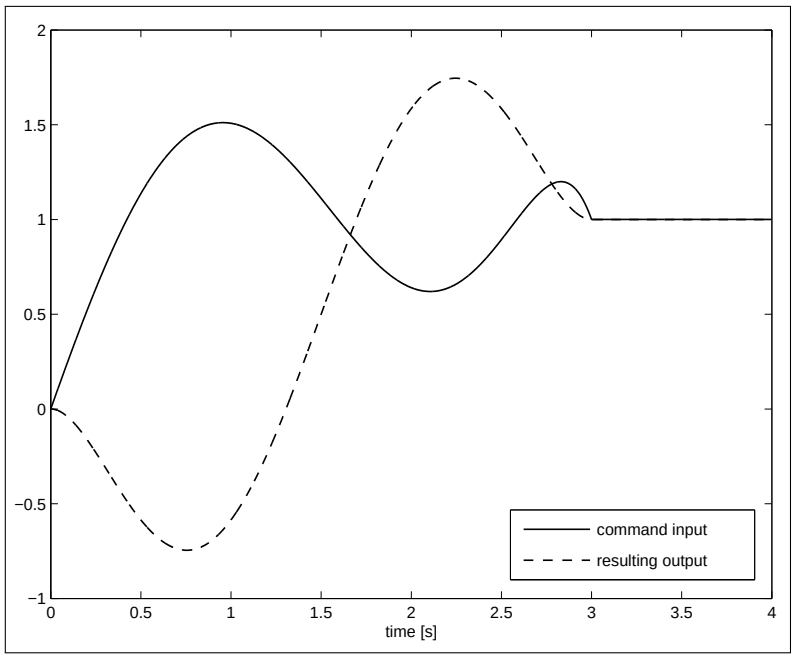


Figure 6: Command input and resulting system output with system (19) and a causal approach without post-actuation

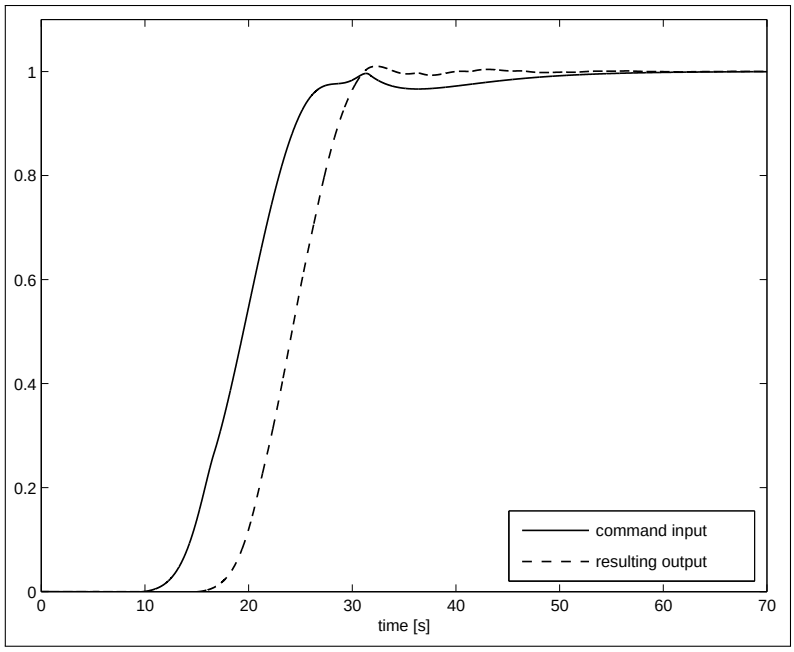


Figure 7: Command input and resulting system output with system (21) with a PID controller

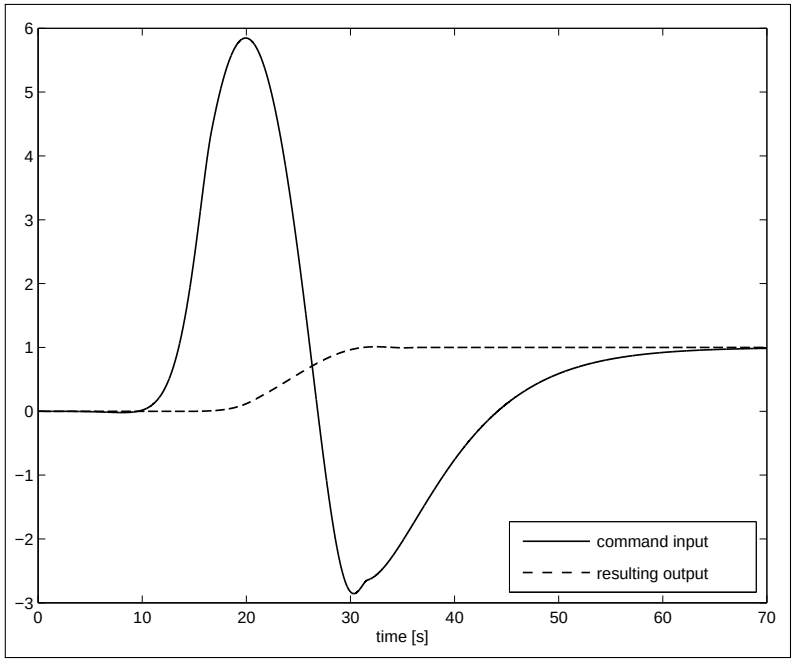


Figure 8: Command input and resulting system output with system (22) with a PID controller

Bibliography

[1] B. C. Kuo, *Automatic Control Systems*, Prentice Hall, Englewood Cliffs (NJ), 1995.

[2] A. Wallen, K. J. Åström, Pulse-step control, *Preprints of the 15th IFAC World Congress on Automatic Control*, Barcelona (Spain), 2002.

[3] A. Visioli, A new design for a PID plus feedforward controller, *Journal of Process Control*, Vol. 14, No. 4, pp. 455-461, 2004.

[4] H. Perez, S. Devasia, Optimal output transitions for linear systems, *Automatica*, Vol. 39, pp. 181-192, 2003.

[5] Q. Zou, S. Devasia, Preview-based optimal inversion for output tracking: application to scanning tunneling microscopy, *Proceedings IEEE International Conference on Decision and Control*, Las Vegas (USA), pp. 79-85, 2002.

[6] D. Iamratanakul, H. Perez, S. Devasia, Feedforward trajectory design for output transitions in discrete-time systems: disk-drive example, *Proceedings of the American Control Conference*, Denver (USA), pp. 3142-3147, 2003.

[7] A. Piazzi, A. Visioli, Minimum-time system-inversion-based motion planning for residual vibration reduction, *IEEE/ASME Transactions on Mechatronics*, Vol. 5, No. 1, pp. 12-22, 2000.

[8] A. Piazzi, A.Visioli, Optimal inversion-based control fo the set-point regulation of nonminimum-phase uncertain scalar systems, *IEEE Transactions on Automatic Control*, Vol. 46, No. 10, pp. 1654-1659, 2001.

[9] M. Benosman, G. Le Vey, Stable inversion of SISO nonminimum phase linear systems through output planning: an experimental application to the one-link flexible manipulator, *IEEE Transactions on Control Systems Technology*, Vol. 11, No. 4, pp. 588-597, 2003.

[10] C. Guarino Lo Bianco, A. Piazzi, A servo control system design using dynamic inversion, *Control Engineering Practice*, Vol. 10, No, 8, pp. 847-855, 2002.

[11] A. Piazzi, A. Visioli, Optimal dynamic inversion based control of an overhead crane, *IEE Proceedings - Control Theory and Applications*, Vol. 149, No. 5, pp. 405-411, 2002.

[12] A. Visioli, A. Piazzi, Improving set-point following performance of industrial controllers with a fast dynamic inversion algorithm, *Industrial Engineering and Chemistry Research*, Vol. 42, pp. 1357-1362, 2003.

[13] A. Piazzi, A. Visioli, A noncausal approach for PID control, *Journal of Process Control*, Vol. 16, pp. 831-843, 2006.

[14] J. W. Polderman, J. C. Willems, *Introduction to Mathematical Systems Theory*, Springer-Verlag, New York, 1998.

[15] D. Pallastrelli, A. Piazzi, Stable dynamic inversion of nonminimum-phase scalar linear systems, *Preprints of the 16th IFAC World Congress on Automatic Control*, Prague (CZ), 2005.

[16] Q. Zou, S. Devasia, Preview-based inversion of nonlinear nonminimum-phase systems: VTOL example, *Proceedings of the IEEE International Conference on Decision and Control*, Paradise Island (The Bahamas), pp. 4350-4356, 2004.

[17] A. Piazzi, A. Visioli, Optimal noncausal set-point regulation of scalar systems, *Automatica*, Vol. 37, No. 1, pp. 121-127, 2001.

[18] A. Piazzi, A. Visioli, Using stable input-output inversion for minimum-time feedforward constrained regulation of scalar systems, *Automatica*, Vol. 41, No. 2, pp. 305-313, 2005.

[19] E. Hansen, G. W. Walster, *Global optimization using interval analysis* - 2nd edition, Marcel Dekker, 2003.

Antonio Visioli
University of Brescia
Dipartimento di Elettronica per l'Automazione
Via Branze 38, I-25123 Brescia, Italy
E-mail: antonio.visioli@ing.unibs.it

Aurelio Piazzi
University of Parma
Dipartimento di Ingegneria dell'Informazione
Parco Area delle Scienze 181A, I-43100 Parma, Italy

Received: January 2, 2007

Antonio Visioli received the Laurea degree in electronic engineering from the University of Parma, Parma, Italy, and the Ph.D. degree in applied mechanics from the University of Brescia, Brescia, Italy, in 1995 and 1999 respectively. His Ph.D. dissertation was on control strategies for industrial robot manipulators. He is currently an Associate Professor of Automatic Control, Department of Electronics for Automation, University of Brescia, Italy. His research interests include industrial robot control and trajectory planning, dynamic-inversion-based control and process control. He is the author or co-author of more than 100 papers in international journals and refereed conference proceedings and he is the author of the book *Practical PID Control* published by Springer. Dr. Visioli is a senior member of IEEE and a member of IFAC and Anipla (Italian Association for the Automation).

Aurelio Piazzi received the Laurea degree in nuclear engineering in 1982 and the Ph.D. degree in system engineering in 1987, both from the University of Bologna, Italy. From 1990 to 1992, he was Research Associate in System Theory, DEIS, University of Bologna. Since 1992 he has been at the University of Parma where he is currently Full Professor of Automatic Control at the Department of Informatics Engineering. His main research interests are in system and control theory and related engineering applications. His recent research activities have focused on optimization and dynamic inversion techniques for autonomous vehicle guidance and for the design of high-performance control systems. Scientific coordinator of bilateral research programs in collaboration with various industries (among them CNR, ENEL, RFI -Ferrovie dello Stato), in 2002 and 2003 he has directed the European Project COOKIES within the EU cluster Eutist-IMV (Integrated Machine Vision) in collaboration with Gruppo Colussi (Perugia) for the artificial vision-based control of food industrial ovens. He is a member of IEEE and SIAM and has published over 90 scientific papers in international journals and conference proceedings.

Author index

Description

International Journal of Computers, Communications & Control (IJCCC) is published from 2006 and has 4 issues per year (March, June, September, December). IJCCC is edited by CCC Publications, powered by Agora University Editing House, Oradea, ROMANIA.

EBSCO Publishing is a licensed partner of Agora University Editing House in publishing and distributing this quarterly international journal in USA, Canada and other countries.

Every issue is published in online format (ISSN 1841-9844) and print format (ISSN 1841-9836). We offer free online access to the full content of the journal `http://journal.univagora.ro`. The printed version of the journal should be ordered, by subscription, and will be delivered by regular mail.

IJCCC is directed to the international communities of scientific researchers from the universities, research units and industry.

IJCCC publishes original and recent scientific contributions in the following fields:

- Computing & Computational Mathematics,
- Information Technology & Communications,
- Computer-based Control

The publishing policy of IJCCC encourages particularly the publishing of scientific papers that are focused on the convergence of the 3 "C" (Computing, Communications, Control).

Topics of interest include, but are not limited to the following: Applications of the Information Systems, Artificial Intelligence, Automata and Formal Languages, Collaborative Working Environments, Computational Mathematics, Cryptography and Security, E-Activities, Fuzzy Systems, Informatics in Control, Information Society - Knowledge Society, Natural Computing, Network Design & Internet Services, Multimedia & Communications, Parallel and Distributed Computing.

The articles submitted to IJCCC must be original and previously unpublished in other journals. The submissions will be revised independently by two reviewers.

IJCCC also publishes:

- papers dedicated to the works and life of some remarkable personalities;
- reviews of some recent important published books.

Also, IJCCC will publish as supplementary issues the proceedings of some international conferences or symposiums on Computers, Communications and Control, scientific events that have reviewers and program committee.

The authors are kindly asked to observe the rules for typesetting and submitting described in *Instructions for Authors*.

Instructions for authors

Papers submitted to the International Journal of Computers, Communications & Control must be prepared using a LaTeX typesetting system. A template for preparing the papers is available on the journal website `http://journal.univagora.ro`. In the template file you will find instructions that will help you prepare the source file. Please, read carefully those instructions.
Any graphics or pictures must be saved in Encapsulated PostScript (.eps) format.
Papers must be submitted electronically to the following address: `ccc@univagora.ro`.
The papers must be written in English. The first page of the paper must contain title of the paper, name of author(s), an abstract of about 300 words and 3-5 keywords. The name, affiliation (institution and department), regular mailing address and email of the author(s) should be filled in at the end of the paper. The last page should include a short bio-sketch and a picture of all the authors. Examples you may find in the previous issues of the journal.
Manuscripts must be accompanied by a signed copyright transfer form. The copyright transfer form is available on the journal website.
When you receive the acceptance for publication you will have to send us:

1. Completed copyright transfer form.
2. Source (input) files.
    - One LaTeX file for the text.
    - EPS files for figures - they must reside in a separate folder.
3. Final PDF file (for reference).
4. A short (maximum 200 words) bio-sketch and a picture of all authors to be included at the end of the article.

One author may submit for publication at most two articles/year. The maximum number of authors is four. The maximum number of pages of one article is 16 (including a bio-sketch). The publishing of a 10 page article is free of charge. For each supplementary page there is a fee of 50 Euro/page that must be paid after receiving the acceptance for publication.
The authors don't receive a printed copy of the journal.
The journal is freely available on `http://journal.univagora.ro`.

Order

If you are interested in having a subscription to "Journal of Computers, Communications and Control", please fill in and send us the order form below:

ORDER FORM		
I wish to receive a subscription to "Journal of Computers, Communications and Control"		
NAME AND SURNAME:		
Company:		
Number of subscription:	Price Euro	for issues yearly (4 number/year)
ADDRESS: City: Zip code: Country: Fax: Telephone: E-mail:		
Notes for Editors (optional)		

1. Standard Subscription Rates for Romania (4 issues/2007, more than 400 pages, including domestic postal cost): 90 EURO.

   2. Standard Subscription Rates for other countries (4 issues/2007, more than 400 pages, including international postal cost): 160 EURO.

For payment subscription rates please use following data:

HOLDER: Fundatia Agora, CUI: 12613360
BANK: BANK LEUMI ORADEA
BANK ADDRESS: Piata Unirii nr. 2-4, Oradea, ROMANIA
IBAN ACCOUNT for EURO: RO02DAFB1041041A4767EU01
IBAN ACCOUNT for LEI/ RON: RO45DAFB1041041A4767RO01
SWIFT CODE (eq. BIC): DAFBRO22

Mention, please, on the payment form that the fee is "for IJCCC".

EDITORIAL ADDRESS:

CCC Publications
Piata Tineretului nr. 8
ORADEA, jud. BIHOR
ROMANIA
Zip Code 410526
Tel.: +40 259 427 398
Fax: +40 259 434 925
E-mail: ccc@univagora.ro, Website: www.journal.univagora.ro

Copyright Transfer Form

To The Publisher of the International Journal of Computers, Communications & Control

This form refers to the manuscript of the paper having the title and the authors as below:

The Title of Paper (hereinafter, "Paper"):

.....

The Author(s):

.....

.....

.....

.....

The undersigned Author(s) of the above mentioned Paper here by transfer any and all copyright-rights in and to The Paper to The Publisher. The Author(s) warrants that The Paper is based on their original work and that the undersigned has the power and authority to make and execute this assignment. It is the author's responsibility to obtain written permission to quote material that has been previously published in any form. The Publisher recognizes the retained rights noted below and grants to the above authors and employers for whom the work performed royalty-free permission to reuse their materials below. Authors may reuse all or portions of the above Paper in other works, excepting the publication of the paper in the same form. Authors may reproduce or authorize others to reproduce the above Paper for the Author's personal use or for internal company use, provided that the source and The Publisher copyright notice are mentioned, that the copies are not used in any way that implies The Publisher endorsement of a product or service of an employer, and that the copies are not offered for sale as such. Authors are permitted to grant third party requests for reprinting, republishing or other types of reuse. The Authors may make limited distribution of all or portions of the above Paper prior to publication if they inform The Publisher of the nature and extent of such limited distribution prior there to. Authors retain all proprietary rights in any process, procedure, or article of manufacture described in The Paper. This agreement becomes null and void if and only if the above paper is not accepted and published by The Publisher, or is withdrawn by the author(s) before acceptance by the Publisher.

Authorized Signature (or representative, for ALL AUTHORS): .....

Signature of the Employer for whom work was done, if any: .....

Date: .....

Third Party(ies) Signature(s) (if necessary): .....

ICCCC 2008
International Conference on Computers, Communications & Control (ICCCC 2008), Băile Felix (Oradea), Romania, 15-17 May 2008

The IEEE 2nd International Conference on Computers, Communications & Control (ICCCC 2008) is organized by Agora University and IEEE Computer Society - Romania Section.
ICCCC 2008 provides a forum for international scientists in academia and industry to present and discuss their latest research findings on a broad array of topics in computer networking and control.
The Program Committee is soliciting paper describing original, previously unpublished, completed research, not currently under review by another conference or journal, addressing state-of-the-art research and development in all areas related to computer networking and control.
The papers will be reviewed by two members of the program committee or other specialists.
The conference proceedings will be available at the conference. Some selected papers will be published in: International Journal of Computers, Communications & Control (IJCCC).

General topics of interest include, but are not limited to, the following:

1. Artificial Intelligence
2. Automata and Formal Languages
3. Computational Mathematics
4. Cryptography and Security
5. Control
6. Economic Informatics
7. E-Activities
8. Fuzzy Systems
9. Information Society - Knowledge Society
10. Natural Computing
11. Network Design & Internet Services
12. Multimedia & Communications
13. P2P Systems and Internet applications
14. Parallel and Distributed Computing

This event is dedicated to the Centenary of *John Bardeen* (1908-1991) the co-inventor of the transistor, a very important element in the development of the computers and the communications.
For more details see www.iccc.univagora.ro