

**Katalin M. Hangos, Rozália Lakner
and Miklós Gerzson**

INTELLIGENT CONTROL SYSTEMS

An Introduction with Examples

Kluwer Academic Publishers

Intelligent Control Systems

An Introduction with Examples

by

Katalin M. Hangos

Department of Computer Science,

University of Veszprém,

Systems and Control Laboratory,

Computer and Automation Research Institute of the Hungarian Academy of Sciences

Rozália Lakner

Department of Computer Science,

University of Veszprém

and

Miklós Gerzson

Department of Automation,

University of Veszprém



KLUWER ACADEMIC PUBLISHERS

DORDRECHT / BOSTON / LONDON

A C.I.P. Catalogue record for this book is available from the Library of Congress.

ISBN 1-4020-0134-7

Published by Kluwer Academic Publishers,
P.O. Box 17, 3300 AA Dordrecht, The Netherlands.

Sold and distributed in North, Central and South America
by Kluwer Academic Publishers,
101 Philip Drive, Norwell, MA 02061, U.S.A.

In all other countries, sold and distributed
by Kluwer Academic Publishers,
P.O. Box 322, 3300 AH Dordrecht, The Netherlands.

Printed on acid-free paper

All Rights Reserved

© 2001 Kluwer Academic Publishers

No part of the material protected by this copyright notice may be reproduced or utilized in any form or by any means, electronic or mechanical, including photocopying, recording or by any information storage and retrieval system, without written permission from the copyright owner

Printed in the Netherlands.

Acknowledgments

With the high popularity and expectations of intelligent control systems in our minds, we felt a great challenge to come up with a textbook in intelligent control systems. That is why we are particularly grateful for all those who have encouraged us to get through: our colleagues, students and families.

The material is based on our intelligent control course for 4th and 5th year information engineers in the University of Veszprém (Hungary) which has been taught successfully for 5 years for more than 100 students. The support of the University, our colleagues and students is gratefully acknowledged.

The inspiring and friendly atmosphere at the Department of Computer Science at the University of Veszprém and that of the Systems and Control Laboratory of the Computer and Automation Research Institute has also contributed to the writing of this book.

Special thanks to Gábor Szederkényi who helped us with all technical and LATEX problems.

Preface

Disciplines are diverging and converging. That is a natural process of science. Diverging is the deeply penetrating characteristic of science, opening knowledge about new phenomena and creating new methods. Convergence emerges by the interaction of disciplines, it serves as a relevant driving force towards new more effective syntheses. Convergence is evoked by the subject itself, i.e. by science-supported solving of practical tasks.

Control of industrial processes is the best example. Physics, chemistry and mechanics join the control of dynamically changing processes and control methods as a result of mathematical system theory. We can enumerate several further relations, economy and sociology, the whole world of the process and the applying human being.

Here stops the university educator in writing a textbook: What are the constituents of the basic knowledge for an engineer to be prepared for intelligent control? What are easily digestible, stemming from earlier courses? Where should his/her own course be ended, hoping that the further studies and especially the diligence and practice of the student enhances all these for enabling to complete the realistic, highly complex tasks of intelligent process modeling, design and control? That means the thorough and, on the other hand, general knowledge of system requirements.

The underlying textbook is the result of several years teaching experience and could not be based on similar course books in the field. The reason is evident: dynamic system analysis and synthesis applied ideas of artificial intelligence in the past few years only. These methods relate to the general methods of representation functional dynamics, e.g. Petri-nets; different methods of handling uncertainty, especially in cases where statistics is not sufficient but human experience has a relevant role, e.g. fuzzy concept. The description of dynamics is more meaningful by

qualitative methods due to discrete changes in the status and consistence of the materials concerned. Basic is the application of rules and logical reasoning in the analysis of phenomena and control operation. Special tools, such as programming languages dedicated for logical reasoning, shells for creating consultation systems in a special field, i.e. expert systems should be added, too.

The convergence of disciplines open a very suitable pedagogical means for examples related to the real life phenomena of those procedures where the student is familiar. By this way the reader receives much better insight into the subject, can understand theoretical concepts by his/her own personal impression that enables the stimulation of further steps outlined a little bit above.

I wish success for the textbook and to the students, started with this initiative!

Tibor Vámos
Member of the Hungarian Academy of Sciences
Computer and Automation Research Institute
Budapest, 21th June, 2001

Contents

Acknowledgments	xiii
Preface	xv
1. GETTING STARTED	1
1. Intelligent control: what does it mean?	2
2. Components of intelligent control systems	3
2.1 Software elements	3
2.2 Users	5
3. The structure and use of the book	6
3.1 The structure of the material	6
3.2 Prerequisites and potential readers	7
3.3 Course variants	8
2. KNOWLEDGE REPRESENTATION	11
1. Data and knowledge	12
1.1 Data representation and data items in traditional databases	12
1.2 Data representation and data items in relational databases	14
2. Rules	15
2.1 Logical operations	15
2.2 Syntax and semantics of rules	18
2.3 Datalog rule sets	19
2.3.1 The dependence graph of datalog rule sets	21
3. Objects	22
4. Frames	26
5. Semantic nets	27
3. REASONING AND SEARCH IN RULE-BASED SYSTEMS	31
1. Solving problems by reasoning	31
1.1 The structure of the knowledge base	32
1.2 The reasoning algorithm	33
1.3 Conflict resolution	36

1.4	Explanation of the reasoning	38
2.	Forward reasoning	38
2.1	The method of forward reasoning	38
2.2	A simple case study of forward reasoning	41
3.	Backward reasoning	44
3.1	Solving problems by reduction	44
3.2	The method of backward reasoning	45
3.3	A simple case study of backward reasoning	48
4.	Bidirectional reasoning	51
5.	Search methods	51
5.1	The general search algorithm	52
5.2	Depth-first search	53
5.3	Breadth-first search	54
5.4	Hill climbing search	55
5.5	A* search	56
4.	VERIFICATION AND VALIDATION OF RULE-BASES	59
1.	Contradiction freeness	60
1.1	The notion of contradiction freeness	60
1.2	Testing contradiction freeness	61
1.3	The search problem of contradiction freeness	63
2.	Completeness	64
2.1	The notion of completeness	64
2.2	Testing completeness	64
2.3	The search problem of completeness	65
3.	Further problems	66
3.1	Joint contradiction freeness and completeness	66
3.2	Contradiction freeness and completeness in other types of knowledge bases	66
4.	Decomposition of knowledge bases	67
4.1	Strict decomposition	68
4.2	Heuristic decomposition	68
5.	TOOLS FOR REPRESENTATION AND REASONING	69
1.	The Lisp programming language	70
1.1	The fundamental data types in Lisp	70
1.2	Expressions and their evaluation	72
1.3	Some useful Lisp primitives	73
1.3.1	The QUOTE primitive	73
1.3.2	Primitives manipulate on lists	74
1.3.3	Assignment primitives	76
1.3.4	Arithmetic primitives	76
1.3.5	Predicates	77
1.3.6	Conditional primitives	79
1.3.7	Procedure definition	81
1.4	Some simple examples in Lisp	82
1.4.1	Logical functions	82
1.4.2	Calculating sums	83

1.4.3	Polynomial value	84
2.	The Prolog programming language	84
2.1	The elements of Prolog programs	85
2.1.1	Facts	85
2.1.2	Rules	87
2.1.3	Questions	87
2.1.4	The Prolog program	88
2.1.5	The declarative and procedural views of a Prolog program	89
2.1.6	More about lists	89
2.2	The execution of Prolog programs	90
2.2.1	How questions work	90
2.2.2	Unification	92
2.2.3	Backtracking	93
2.2.4	Tracing Prolog execution	94
2.2.5	The search strategy	95
2.2.6	Recursion	96
2.3	Built-in predicates	96
2.3.1	Input-output predicates	97
2.3.2	Dynamic database handling predicates	97
2.3.3	Arithmetic predicates	98
2.3.4	Expression-handling predicates	98
2.3.5	Control predicates	99
2.4	Some simple examples in Prolog	99
2.4.1	Logical functions	99
2.4.2	Calculation of sums	100
2.4.3	Path finding in a graph	101
3.	Expert system shells	103
3.1	Components of an expert system shell	104
3.2	Basic functions and services in an expert system shell	105
6.	REAL-TIME EXPERT SYSTEMS	109
1.	The architecture of real-time expert systems	110
1.1	The real-time subsystem	111
1.2	The intelligent subsystem	113
2.	Synchronization and communication between real-time and intelligent subsystems	114
2.1	Synchronization and communication primitives	114
2.2	Priority handling and time-out	115
3.	Data exchange between the real-time and the intelligent subsystems	116
3.1	Loose data exchange	117
3.2	The blackboard architecture	119
4.	Software engineering of real-time expert systems	121
4.1	The software lifecycle of real-time expert systems	122
4.2	Special steps and tools	125
7.	QUALITATIVE REASONING	127

1.	Sign and interval calculus	128
1.1	Sign algebra	129
1.2	Interval algebras	130
2.	Qualitative simulation	132
2.1	Constraint type qualitative differential equations	132
2.2	The solution of QDEs: the qualitative simulation algorithm	138
2.2.1	Initial data for the simulation	138
2.2.2	Steps of the simulation algorithm	139
2.2.3	Simulation results	142
3.	Qualitative physics	145
3.1	Confluences	145
3.2	The use of confluences	147
4.	Signed directed graph (SDG) models	148
4.1	The structure graph of state-space models	148
4.2	The use of SDG models	151
8.	PETRI NETS	153
1.	The Notion of Petri nets	154
1.1	The basic components of Petri nets	154
1.1.1	Introductory examples	154
1.1.2	The formal definition of Petri nets	162
1.2	The firing of transitions	162
1.3	Special cases and extensions	165
1.3.1	Source and sink transitions	165
1.3.2	Self-loop	165
1.3.3	Capacity of places	166
1.3.4	Parallelism	168
1.3.5	Inhibitor arcs	172
1.3.6	Decomposition of Petri nets	175
1.3.7	Time in Petri nets	176
1.4	The state-space of Petri nets	177
1.5	The use of Petri nets for intelligent control	178
2.	The analysis of Petri nets	178
2.1	Analysis Problems for Petri Nets	179
2.1.1	Safeness and Boundedness	179
2.1.2	Conservation	179
2.1.3	Liveness	180
2.1.4	Reachability and Coverability	180
2.1.5	Structural properties	180
2.2	Analysis techniques	181
2.2.1	The reachability tree	181
2.2.2	Analysis with matrix equations	186
9.	FUZZY CONTROL SYSTEMS	191
1.	Introduction	191
1.1	The notion of fuzziness	191
1.2	Fuzzy controllers	192
2.	Fuzzy sets	192

2.1	Definition of fuzzy sets	192
2.2	Operations on fuzzy sets	200
2.2.1	Primitive fuzzy set operations	201
2.2.2	Linguistic modifiers	205
2.3	Inference on fuzzy sets	208
2.3.1	Relation between fuzzy sets	209
2.3.2	Implication between fuzzy sets	211
2.3.3	Inference on fuzzy sets	214
3.	Rule-based fuzzy controllers	215
3.1	Design of fuzzy controllers	216
3.1.1	The input and output signals	216
3.1.2	The selection of universes and membership functions	217
3.1.3	The rule-base	219
3.1.4	The rule-base analysis	220
3.2	The operation of fuzzy controllers	223
3.2.1	The preprocessing unit	223
3.2.2	The inference engine	223
3.2.3	The postprocessing unit	225
10.G2:	AN EXAMPLE OF A REAL-TIME EXPERT SYSTEM	227
1.	Knowledge representation in G2	228
2.	The organization of the knowledge base	230
2.1	Objects and object definitions	231
2.2	Workspaces	232
2.3	Variables and parameters	233
2.4	Connections and relations	234
2.5	Rules	235
2.6	Procedures	237
2.7	Functions	238
3.	Reasoning and simulation in G2	239
3.1	The real-time inference engine	239
3.2	The G2 simulator	240
4.	Tools for developing and debugging knowledge bases	241
4.1	The developers' interface	241
4.1.1	The graphic representation	241
4.1.2	G2 grammar	242
4.1.3	The interactive text editor	242
4.1.4	The interactive icon editor	243
4.1.5	Knowledge base handling tools	244
4.1.6	Documenting in the knowledge base	245
4.1.7	Tracing and debugging facilities	246
4.1.8	The access control facility	247
4.2	The end-user interface	247
4.2.1	Displays	247
4.2.2	End-user controls	248
4.2.3	Messages, message board and logbook	249
4.3	External interface	250

Appendices	251
A- A BRIEF OVERVIEW OF COMPUTER CONTROLLED SYSTEMS	251
1. Basic notions in systems and control theory	251
1.1 Signals and signal spaces	252
1.2 Systems	252
2. State-space models of linear and nonlinear systems	253
2.1 State-space models of LTI systems	254
2.2 State-space models of nonlinear systems	254
2.3 Controllability	255
2.4 Observability	256
2.5 Stability	257
3. Common functions of a computer controlled system	258
3.1 Primary data processing	258
3.2 Process monitoring functions	260
3.3 Process control functions	260
3.4 Functional design requirements	262
4. Real-time software systems	262
4.1 Characteristics of real-time software systems	262
4.2 Elements of real-time software systems	264
4.3 Tasks in a real-time system	264
5. Software elements of computer controlled systems	268
5.1 Characteristic data structures of computer controlled systems	268
5.1.1 Raw measured data and measured data files	269
5.1.2 Primary processing data file	270
5.1.3 Events data file	270
5.1.4 Actuator data file	271
5.2 Typical tasks of computer controlled systems	272
5.2.1 Measurement device handling	272
5.2.2 Primary and secondary processing	272
5.2.3 Event handling	272
5.2.4 Controller(s) and actuator handling	273
B- THE COFFEE MACHINE	275
1. System description	275
2. Dynamic model equations	277
2.1 Differential (balance) equations	278
2.2 System variables	279
References	281
Index	289
About the Authors	301

Chapter 1

GETTING STARTED

Intelligent control is a rapidly developing, complex and challenging field with great practical importance and potential. It emerged as an interdisciplinary field of computer controlled systems and artificial intelligence (AI) in the late seventies or early eighties when the necessary technical and theoretical infrastructure in both computer science and real-time computation techniques became available.

A great deal of interest has been shown in learning more about intelligent control by a wide audience. It has been a challenging and popular course subject for both graduate and undergraduate students of various engineering disciplines. At the same time there is a growing need amongst industrial practitioners to have textbook material on the subject readily to hand.

Because of the rapidly developing and interdisciplinary nature of the subject, the information available is mainly found in research papers, intelligent control system manuals and – last but not least – in the minds of practitioners, of engineers and technicians in various fields. There are a few edited volumes consisting of research papers on intelligent control systems [1], [2]. Little is known and published about the *fundamentals and the general know-how in designing, implementing and operating intelligent control systems*. Therefore, the subject is suitable mainly for elective courses on an advanced level where both the material and the presentation could and should be flexible: a core basic material is supplemented with variable parts dealing with the special tools and techniques depending on the interest and background of the participants.

1. INTELLIGENT CONTROL: WHAT DOES IT MEAN?

The notion of intelligent control systems is based on a joint understanding of the notions of "control systems" and "intelligent systems". Both of the above notions have undergone a strong development and have been the subject of disputes and discussions (see e.g. [3]). Therefore we shall restrict ourselves to practical, engineering type definitions of both, in describing the subject matter of this book.

Control systems assume the existence of a dynamic system to be controlled, that is an object the behaviour of which is time-dependent behaviour and which responds to the influences of its environment described by the so called input signals by output signals. The control system then senses both input and output and designs an input that achieves a pre-defined control aim.

Control systems are most often realized using computers, and in these cases we talk about *computer-controlled systems*. A computer-controlled system is by nature a real-time software system. Its software architecture contains standard data structures and tasks operating thereon. These include the following:

- *data structures*: raw measured data, measured data, events, etc.
- *tasks*: measurement device handling, primary processing, event handling, etc.

Appendix A gives a detailed description of the most important terms and notions in systems and control theory, as well as the software structure of a computer controlled system.

The notion of *intelligence* in the sense of *artificial intelligence* [4]-[8] is the other ingredient in the term "intelligent control systems". The notion of *intelligence* in itself has been a subject of permanent discussion for a long time and *artificial intelligence* is understood as "computer-aided intelligence", that is intelligence produced by computers.

The engineering type definition of artificial intelligence can be best understood if one recalls the elements of a problem for which we think we need a clever or "intelligent" solution. It is intuitively clear that easy or trivial tasks do not need a clever solution, just – perhaps – hard work. On the other hand, clever or intelligent solutions exhibit at least some non-trivial, surprising or unusual element, approach or other ingredient [9]. Therefore, one may say that an *intelligent method solves*

- a *difficult (non-trivial, complex, unusually large or complicated) problem*

- *in a non-trivial, human-like way.*

Furthermore, we can identify another basic characteristic of intelligent methods if we follow the idea of the engineering type definition above. The basic difference between the human and the machine way of solving difficult problems is that humans prefer to use clever heuristics over mechanistic exhaustive "brute force" approaches. The presence of *heuristics* is one of the key characteristics of intelligent methods.

To summarize we can say that *intelligent control systems are computer-controlled systems where at least part of the control tasks performed require intelligent methods.*

2. COMPONENTS OF INTELLIGENT CONTROL SYSTEMS

Every object with some kind of intelligence exhibits a quite complex and sophisticated structure: think of the biological structure of our nervous system controlled by our brain. Similarly, intelligent control systems have special components which are necessary to carry out control in an intelligent way. Most of the software elements of an intelligent control system perform its control function but some special elements serve its users, who come from various backgrounds and have varying academic qualifications.

2.1 SOFTWARE ELEMENTS

As we have already seen before, intelligent control systems are computer controlled systems with intelligent element(s) [10]. This implies that Neuman's principle applies to these systems: they have separate elements for the inherently passive, *data type* part and the active, *program type* part.

In traditional software systems, like in computer controlled systems, the data type elements are usually organized in a *database* while the active elements are *real-time tasks*. Tasks share the data in the database and a special task, the *database manager* is responsible for the resource management and the consistency of the data base.

This separation is clearly visible on the software structure of a computer controlled system described in details in section 5. in Appendix A.

Clearly not every intelligent system obeys Neuman's principle. Our brain, for example, works in a distributed manner, where every neuron has processing functions and stores data as well by connecting to other neurons.

The intelligent software systems that obey Neuman's principle are called *knowledge-based* systems. In intelligent software systems one can also find elements of the data and program type, therefore they are all knowledge-based systems. These elements, however, are given other special names as compared to traditional software systems.

The basic elements of a knowledge-based system are depicted in Fig. 1.1.

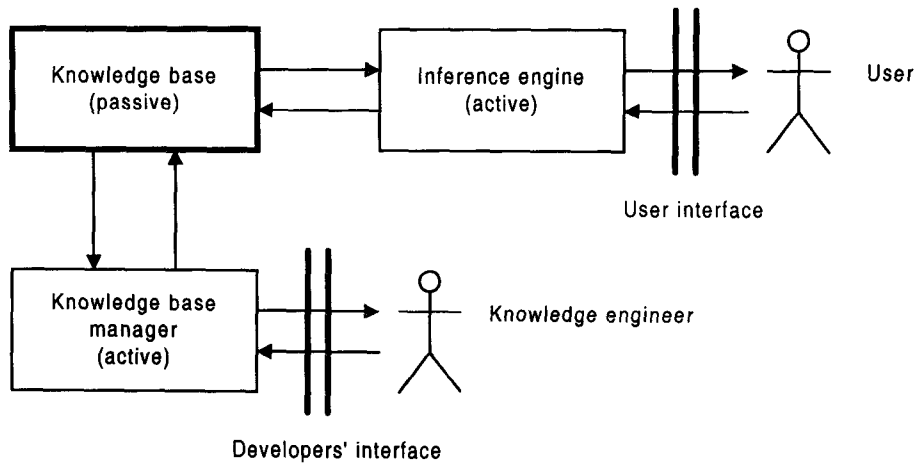


Figure 1.1. The structure of knowledge-based systems

We can see the following active and passive elements:

1. *Knowledge base*

The database of a knowledge-based system is called the knowledge base. There is, however, a substantial difference between a database with data entirely passive and a knowledge base where the relationships between the individual data elements are much more important. We shall learn more about the similarities and differences between data and knowledge bases in Chapter 2.

2. *Inference engine*

The inference engine of a knowledge-based system is its processing (program) element. It uses the content of the knowledge base to derive new knowledge items using the process of *reasoning*. Reasoning in rule-based expert systems is the subject of a separate chapter, Chapter 3.

There can be more than one inference engine in a knowledge-based system, in the same way as there are multitasking traditional software systems.

3. *Knowledge base manager*

Similarly to the database manager, the knowledge base manager of a knowledge-based system performs the resource and consistency management of the knowledge base. However, this task is much more difficult than that of the database manager's, because the relationships between knowledge items are much more complex. As it is shown in Chapter 4 even checking the completeness and contradiction freeness of a rule-based knowledge base is computationally hard.

There is a special, important and widely used special type of knowledge-based systems where the knowledge is collected from an expert in a specific application domain. Such a knowledge-based system in a specific domain is called an *expert system*. If, in addition, the knowledge base contains data items and logical relationships between them expressed in the form of *rules* we speak about a *rule-based expert system* [11].

2.2 USERS

There are two principally different types of users in any knowledge-based system and their roles, qualification and user privileges are different.

1. *Knowledge engineer*

A knowledge engineer is a person with a degree in computing, software engineering, programming or alike with specialization in intelligent systems. The design, implementation, verification and validation of a knowledge-based system is done by knowledge engineers. Ideally, they should have an interdisciplinary background knowing both knowledge-based systems technology and the application field in which the knowledge-based system is being used. In the case of intelligent control systems, a knowledge engineer should be familiar with the basic notions and principles of computer controlled systems as well.

Knowledge engineers use the so called *developers' interface* which is designed to work directly with the knowledge base manager of the knowledge-based system. Through this interface high privilege tasks, such as changing the structure and content of the knowledge base and other knowledge base management tasks can be carried out.

2. *User*

A knowledge-based system is most often used via the so called *user*