A few discussion on expert system development for electrical power systems-optimization of an inverter-motor of a railway traction chain

Djamel Fezzani, Hubert Piquet, Henri Foch, Philippe Nogaret

A few discussion on expert system development for electrical power systems-optimization of an inverter-motor of a railway traction chain

D. Fezzani[1], H. Piquet[1], H. Foch[1], and Ph . Nogaret[2]
[1] Laboratoire d'Électrotechnique et d'Électronique Industrielle[a], I.N.P. Toulouse, E.N.S.E.E.I.H.T., 2 rue Charles Camichel, 31071 Toulouse Cedex, France
[2] Alstom Transport SA, rue du Docteur Guinier BP 4, 65600 Séméac, France

Abstract. This paper deals with the automatic design of static converters and electrical machines. Facing the multitude of such devices (converters–machines), the choice of the appropriate converter or machine is a delicate problem. A high level of expertise is required to determine the best structure for the considered application. This choice could be rapidly made if one had a system which incorporates a great knowledge of converters, machines and control methods. Firstly, the authors review the realized studies, then they propose this methodology for a concrete case in order to optimize an inverter–motor of a railway traction chain. Constraints are described in terms of input and output performances (losses, current, ...). These data are analyzed by using design rules and methods which allow the decrease of the manufacture cost of the inverter–motor. Rules and methods are implemented on a workstation by means of an "expert system shell" Smeci. The article proposes several tracks which involve organization, methodology, confirmation and implementation of a tool supporting a method. This process of implementation is the most efficient to fit into a constant process of improvement which is supported by the return on experience and by the evolution of the state of the art.

PACS. 02.60.x Numerical approximation and analysis – 02.90.+p Other topics in mathematical methods in physics – 01.50.Kw Testing theory and techniques

1 Introduction

Our work concerns the definition of a design methodology for power electronics systems based on previous concrete experience. The proposed methodology is considered as a fundamental step toward the design of computer–based systems that focus on the interaction of electrical and machine components. It requires therefore an analysis of the design process and to make the share of the different components of the design (rules, practice, norms, calculation methods, various models, *etc.*) in order to propose a systematic method allowing the designers of high power systems to answer efficiently and quickly the different calls for bids.

The aim of this methodology is to elaborate a global approach which is able, according to the cases and the degree of deepening of the design, to manipulate design rules as well as optimize the process or achieve direct calculation in a unique environment software.

This method should not be limited to help in the specification analysis and the choice of the best devices, but it should be able to help also, in the design and dimensioning of the converters, machines and the association of converter–machine.

Figure 1 presents the basic structure of the considered railway traction chain using an asynchronous motor fed by a voltage source inverter.

In order to fulfil the requirements of the specifications, the problem is to construct an expert system to justify firstly its feasibility in an industrial area, by calculating thermal and electrical constraints of the inverter. The second objective consists in a judicious choice of the command and control law of the inverter to optimize its electrical and thermal constraints (losses) and, consequently, to reduce the manufacture cost of the inverter–motor of the railway traction chain.

Expert systems have been applied to solve a wide range of problems in domains such as engineering, computer science and education. Within each domain, they have been used to solve problems of different types. Types of problems involve design (of electrical systems) [1–7], survey and diagnosis (of complex systems) [8–10], control (of industrial processes) [11] and synthesis (of the family of direct converters such as choppers, inverters and rectifiers) [12].

[a] UMR-CNRS 5828

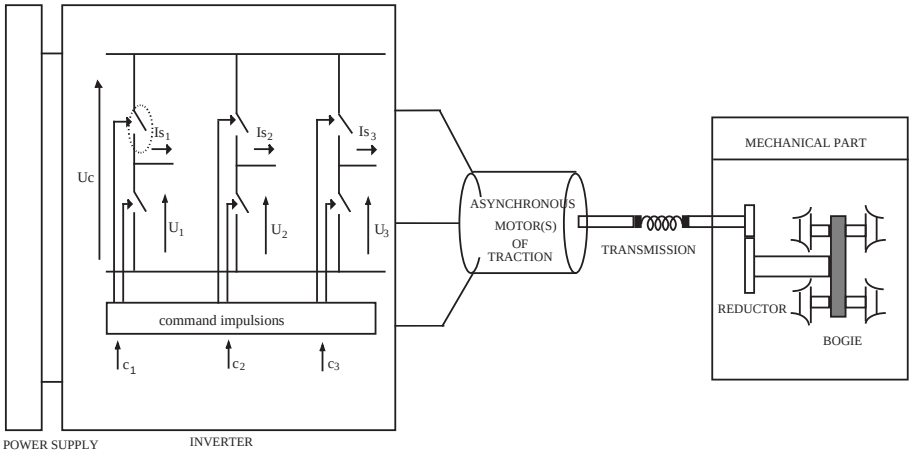

Fig. 1. Description of the studied railway traction chain.

Developments have been led by using the object–oriented programming (OOP) language available in the expert system shell SMECI [13]. The following supported object functionalities are [13,14]:

- the definition of object classes and the creation of instances;
- inheritance of slots, values, and methods;
- overriding of inheritance for slots, values and methods;
- message passing to objects;
- easy modification of an application by simple addition of new classes.

After having presented motivations and interest shown for this new type of programming, the authors will go into more detail on how the expert system is designed and written. First the basic methodology of the realization of the expert system will be reviewed, then they will discuss how suitable problems for the expert system are chosen and the system is developed.

2 Realization methodology of our expert system

The aim of this section is to present the approach which has been used to develop our expert system. This approach is decomposed into several phases which involve the analysis of the human expert step and the identification of the expertise. These stages are highly interrelated and interdependent. An iterative process continues until the software consistently performs at an acceptable level. Figure 2 indicates the most important steps which are essentially the model development and the methodology of systems analysis as well as the main return paths.

2.1 Identification phase

Identification is the requirement analysis step which is carried out in traditional software development. It involves a formal task analysis to determine the external requirements, from the input and output, setting where the program will be used and determining the user. The participants, the problems, the objectives and the resources need to be clearly identified at this stage. Identification phase includes the following steps:

1. firstly, it concerns the identification of the different participants: human specialists as well as methods or tools that they utilize. The simplest task at this stage is, without a doubt, to start from the areas (motor, inverter, *etc.*) to make this analysis;
2. then, it concerns the identification of the progress of the solution between these various actors implied in the resolution of the problem, by trying to localize the know–how for a formalization in terms of rules;
3. finally, it concerns the identification of the places at which new proposals could be made.

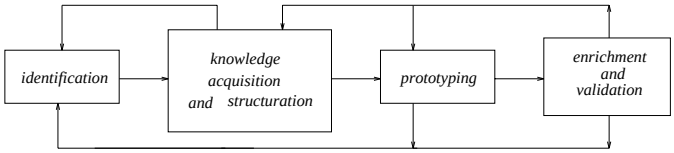

Fig. 2. Development cycle of an expert system.

2.2 Knowledge acquisition and structuring phase

The information that is collected at the first phase allows the undertaking of the conceptualization stage which involves the designing of the proposed program to ensure that the specific interactions and relationships in the problem domain are understood and defined.

This step of knowledge representation as well as the next step which concerns the reasoning structuring, is based on the exchanges undertaken between the experts in the traction domain. These two steps have been particularly long and delicate. Correctly and completely describing the expert's problem solving logic is difficult because experts usually do not know exactly how they reach

a decision and are therefore, often unable effectively to verbalize their own problem solving process [15,16]. In our approach, this task is typically created through frequent and intensive interview sessions with railway traction chain specialists.

In the course of these interview, we have questioned experts and exploited available documents of some indicative businesses. Available documents have been electrical diagrams, operation notes, *etc.* We have used these documents, and especially the interviews reports to discuss with several different experts about their steps facing a problem. In the first instance, these questions concern objects that they manipulate, relationships between these objects, their manner to solve a problem, *etc.* It is around these concepts that we built the knowledge base of the expert system.

The knowledge base concerns the recording of a great number of rules or knowledge modules, academic and stemming from the personal experience which involve verified theories, facts, experiences and all the know–how of the domain expert [17,16,18].

The knowledge acquisition term is frequently interpreted as an extraction of knowledge. It is completely a false image. The conclusion which emerges from interviews with the experts is not directly used, so it is necessary to have a good structuring of the knowledge base.

2.3 Architecture and prototyping

This step involves the organization of the key concepts, subproblems and information flow into formal representations. In fact, the program is designed at this stage. It is often useful to group or modularize the knowledge collected upto now, so we proceed such as:

1. in the first part, we associate each subpart of the railway traction chain with a module which models a part of the problem, then we must ensure that these modules ignore the progress step of the problem. Then, a punctual improvement of the design process will be realized by modifying or by replacing one or several modules;
2. in the second part, we must establish a leader kind which pilots the different modules. This will be charged with distributing the tasks between modules, to recuperate the work realized by these modules and to make a synthesis of the results. This leader kind does not have to be restricted to juxtapose these modules but, it must be able to authorize them to cooperate intelligently.

2.3.1 Definition of an expert module

Taking into account the criteria stated above, we have chosen a decomposition of the railway traction chain into its constitutive elements. This representation leads us to define and characterize an expert module as an entity which implements the "local–knowledge"; it is able, once its specifications are given, to achieve the design of the subpart of the system with which it is associated. An expert module is defined as follows:

- it is associated with one of the subparts of the railway traction chain;
- it realizes the design of this element;
- it partially proceeds to the analysis and to the design of the railway traction chain;
- it calls different types of knowledge such as:
  - calculation formula, numerical programs or graphs;
  - knowledge acquired by experience, often expressed in the form of rules.

2.3.2 Description of an expert module

Each module associated with the different components that constitute the railway traction chain is characterized by the:

- *output* constituted by a set of parameters to be determined. In the case of the *inverter* module, it concerns the determination of the electrical constraints (peak–to–peak values, rms, ... of current and voltage), losses and the definition of firing signals related to any switch;
- *constraints* containing the criteria fixed by the user;
- input *data* concerning the operating conditions. These data can be introduced by means of questions to the user or can be extracted from the *output* reached by other modules;
- *informations* where all the supplementary useful data are stored. These are used, once the module has finished working, to evaluate the obtained results;
- capability to call other expert modules;
- possibility to communicate with external software and, consequently, to use already developed programs.

2.3.3 Functioning of an expert module

The main function of the module is therefore, from its *constraints* and its *data*, to determine the values presented in its *output*. For that purpose, the module calls first its own resources (constituted by a set of rules and methods), otherwise it calls external software, or other expert modules when it is necessary. Figure 3 summarizes the structural and the functional representation shared by all the modules.

2.4 Enrichment and the validation phase

The last stage, enrichment and validation, involves considerably more than finding and fixing syntax errors. It covers the verification of the consistency and completeness of the rules, the ability of the control strategy to consider information in the order that corresponds to the problem solving process, testing guides reformulation of concepts, redesign of representations and other refinements. This

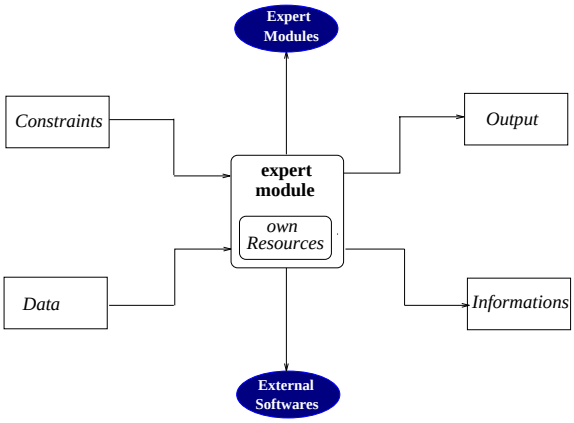

Fig. 3. Structural and functional representation of an expert module.

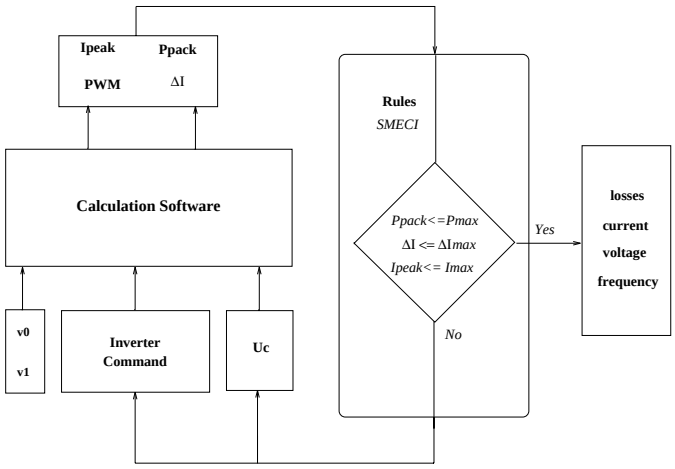

Fig. 4. Optimization procedure.

period is notably difficult; it consists of establishing that the system meets the intended original goal. This can be determined from the program accuracy that is determined from comparisons with the anterior studies or the results given by human experts.

3 Feasibility of the expert system in the traction area

To test the feasibility of the expert system in the traction area, a typical approach is proposed as shown in Figure 4.

The aim is to reduce the manufacture cost of the inverter–motor, by optimizing the electrical and thermal constraints at the inverter level. In Section 2.3.2, we have pointed out that the expert system is characterized by the possibility to communicate with external software. In the considered case, our expert system calls an external calculation software (designed by the TGL–ED [1] team).

Note that this optimization constraint will be undertaken on the integral speed variation range.

The expert system proposes a set of values for the parameters which are usually chosen by human experts. These data are transfered to the simulation program which in turn computes the electrical and thermal stresses of the equipment. Once received by the expert system, these results are once again used to propose new values of the parameters. This iterative process is continued until satisfactory stresses are obtained. The optimization process of the expert system is implemented by means of expert rules (of Sect. 4.2.2).

The advantage of our approach resides in the fact that the system is able to pilot existent external programs which allows the best re–use of code. In addition, its modular structure allows its enrichment by simple addition of rules to reply to other situations which are not currently anticipated.

[1] R&D Service of Alstom Transport, Site of Tarbes, France

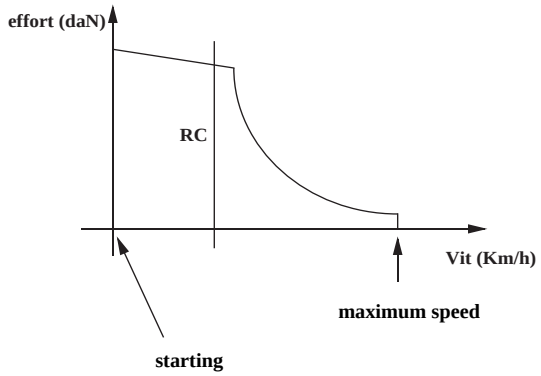

Fig. 5. Traction effort as a function of speed.

3.1 Hypothesis of the study

In order to calculate the voltage source inverter associated with the asynchronous machine, it is necessary to fix some data. It is necessary to define electrical parameters and mechanical configuration of the motor and additionally, it is important to choose the semi–conductors of the inverter, and other parameters (number of switches in parallel by inverter leg, number of motors, *etc.*). Finally, it is necessary to initialize a command for the inverter. To test the feasibility of the proposed approach, the study is carried out under the following assumptions:

- the number of semi–conductors in parallel by inverter leg is fixed equal to 2;
- the optimization will be made for a given motor.

3.2 Specifications

The specifications to be respected are described by the curves expressing the traction effort as a function of locomotive speed (Fig. 5).

3.3 Constraints

Constraints must be specified from the start by the user; they are defined by the:

- peak–to–peak current allowable for one IGBT: I_{max};
- lowest value of DC voltage allowable for the inverter: U_{min};
- maximum value of DC voltage allowable for the inverter: U_{max};
- maximum values of losses allowable for a pack (IGBT + Diode) of the inverter leg: P_{max};
- maximum value of frequency for the intersective asynchronous PWM (see definition in the next section): f_{max};
- lowest value of frequency for the intersective asynchronous PWM (see definition in the next section): f_{min};
- maximum value of gap peak–to–peak inverter current admitted between two changes of PWM: ΔI_{max}.

3.4 Constraints for the PWM

The asynchronous motor has to be fed by the source voltage inverter at nominal flux of the air gap from the stop until the nominal speed. Beyond this speed, the stator voltage reaches its nominal value (full wave operation) and the control of the air gap flux will be difficult.

The command of the asynchronous motor at nominal flux implies respect of the approached formula:

$$\frac{U_s}{f_s} = \frac{U_{s_{nom}}}{f_{s_{nom}}} = k\phi_{m_{nom}} \quad (1)$$

U_s and f_s are respectively the interlinked voltage and the stator frequency.

During starting of the asynchronous motor, the adjustment of the voltage U_s is accomplished by a PWM command law of the inverter. For that purpose, four control modes are distinguished as will be presented in the following.

3.4.1 Intersective asynchronous PWM

The carrier frequency is constant and slightly lower than the maximum of switching frequency of the inverter voltage to limit semi–conductors switching losses.

This monitoring mode is used during the first instants of the starting of the motor. The electromagnetic torque ripple has a small amplitude. It is necessary to specify the speed which delimits the utilization of this control strategy, as well as its switching frequency f. This same frequency must satisfy the following relationship:

$$f_{min} \leq f \leq f_{max}. \quad (2)$$

Figure 6 puts in a prominent position the passing speed of the intersective asynchronous PWM; in this case, the speed corresponds to that of full wave pass (which will be defined later).

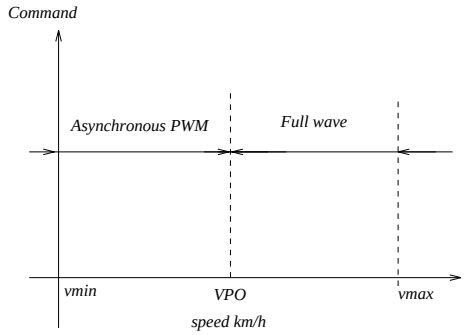

Fig. 6. Initialization of the asynchronous PWM.

3.4.2 Intersective synchronous PWM

The carrier frequency corresponds to a multiple of the motor supply frequency. This characterization allows a good control of the harmonic currents. However, this mode is authorized until the stator fundamental voltage reaches approximately 70% of the nominal value.

Intersective synchronous PWM are characterized by a speed and a number of cuts.

3.4.3 Precalculated synchronous PWM

The number of the calculated angles can vary gradually from 5 to 1. With the different degrees of freedom, some of the harmonic frequencies of the stator voltage are able to be eliminated.

This mode allows the full wave pass and conversely, without an important variation of the motor torque.

3.4.4 Full wave

The inverter can not allow the adjustment of the amplitude of the stator voltage. The interlinked voltage is: $U_{s_{nom}} = 0.78\, U_c$. In these conditions, the inverter switching losses are very reduced.

Nota:

The user can interpose only for choosing the intersective asynchronous PWM, the choice of the other PWM is automatically made by the intermediary of SMECI!

3.5 Major variables to be optimized

For a given data set, the system is able to deliver the electrical and mechanical quantities. For the optimization process, we are interested more particularly in the variables which are submitted to the decision criteria:

- pack losses (IGBT + Diode) for the inverter leg: P_{pack};
- peak current by phase of the inverter: I_{peak};
- gap inverter peak current between two changes of PWM: ΔI.

3.6 Criteria

A solution is considered as optimal when the criteria listed below are reached:

- $P_{pack} \leq P_{max}$;
- $I_{peak} \leq I_{max}$;
- $\Delta I \leq \Delta I_{max}$.

4 Implementation of the expert system prototype

4.1 Formalization of the facts

An expert system is most efficient if the user finds the notions that it has the habit to manipulate. So, it is important to find in the organization of the expert system the structuring of the railway traction chain as previously defined.

The description of the design modules uses the object–oriented programming (OOP) representation [17,19,20,18,21–23].

In this representation, a module is described by using basic classes: *output*, *constraints*, *data* and *informations*, as presented in Section 2.3.2. Thus, the analysis of each module leads to the creation of sub–classes which implement its specific properties. The tree of Figure 7 presents the hierarchy of categories or classes associated with the railway traction chain.

Slots are defined, at the level of each class, to represent object properties. Object slots are, of course, destined to store values.

Thanks to this definition, the functional decomposition dedicated to a specific module (*traction* module) charged with the cooperation between modules (inverter, motor, mechanical part, ...) has been easily implemented. Figure 8 shows categories (classes), objects and slots that are associated with the *traction* module. For example, the class "TRAConstraints" describes the constraints of the *traction* module. Each *instance* (object) of this class has *slots* for the maximum pack losses, lowest and maximum DC voltage, and maximum peak–to–peak current as mentioned in Section 3.3.

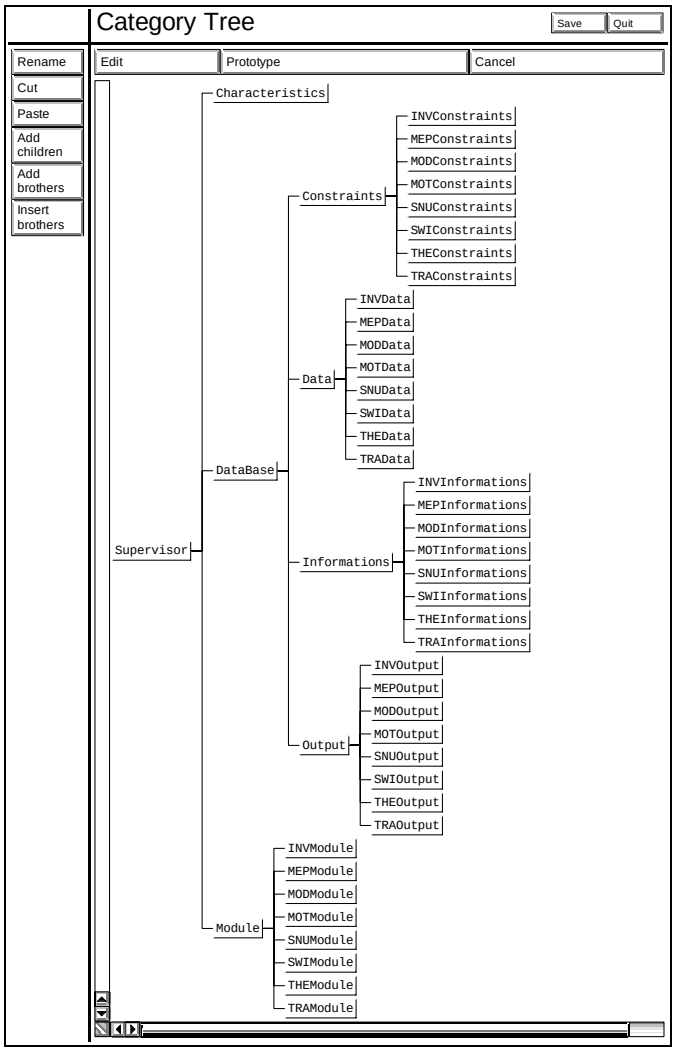

Fig. 7. Object implementation of the railway traction chain.

4.2 Formalization of the declarative knowledge

The representation of the declarative knowledge uses mainly production rules.

4.2.1 Rules

One way of expressing declarative knowledge is through a set of rules. It is a pure declarative representation (very general). The knowledge is represented by a great number of simple deduction rules. These rules are assertions in the form of implications. Each rule represents a quantum of knowledge. The rules allow the description of the process used to make progress towards the resolution of the problem: they handle the slots which describe the objects and release methods.

The rules are written in quasi–natural language and use the predicate logic which allows the utilization of variables and quantificators, giving them a generic character. The syntax used for their representation is:

if Condition(S) then Action(S).

Thus, rules are relationships rather than instructions. Note that this is different to the IF–THEN structure used by procedural programming languages.

Rules are organized in rule bases (see definition in Sect. 4.2.3), that are associated with tasks [3] which are used by Smeci to control the operation of the inference engine and can be considered as classical subroutines.

4.2.2 Different types of rules

Although the formalism adopted for their representation is unique, all rules do not produce the same effects on the

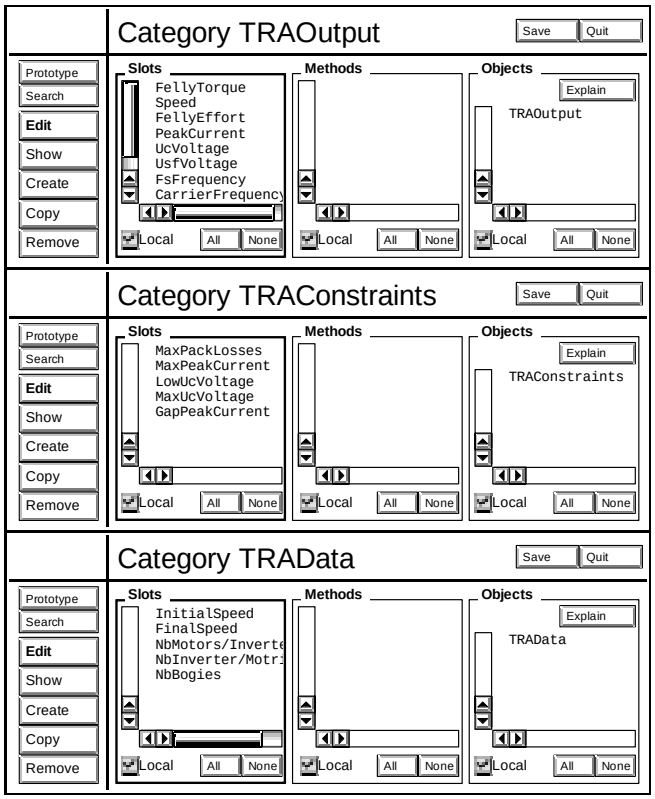

Fig. 8. Categories and slots associated with the traction module.

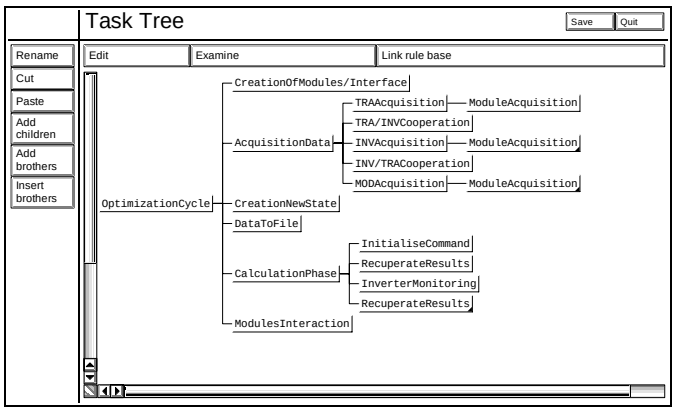

Fig. 9. Task's tree.

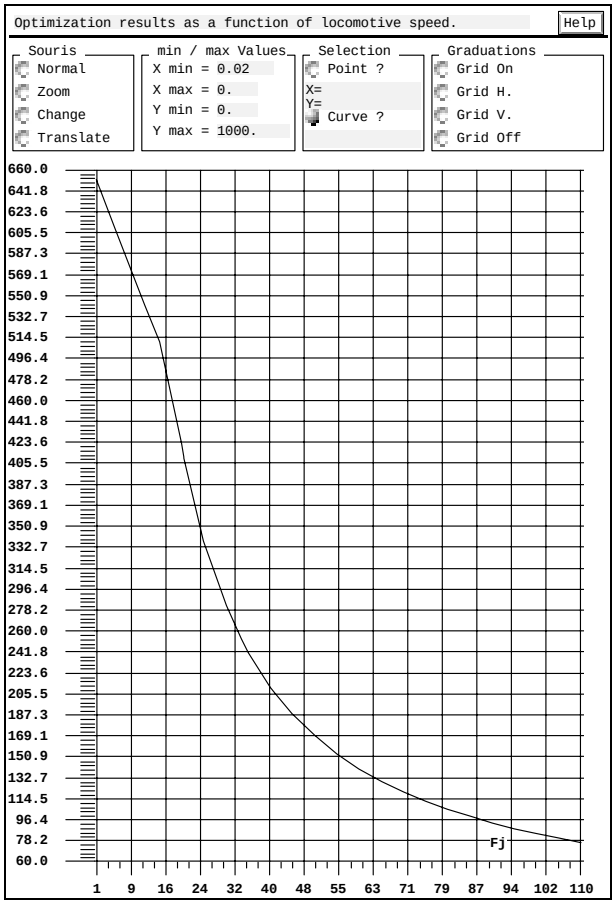

Fig. 10. Specifications: traction effort as function of speed.

resolution of the problem. In this approach, the following major categories can be distinguished:

1. **Hypothesis:** They are the rules that allow the orientation of the process's evolution. They allow also the generation of branches to undertake a parallel reasoning on several states. This fundamental characteristic leads to several solutions for the same problem.
2. **Impasses:** They are the rules that detect if a hopeless path is being investigated, because some conditions will not be able to be filled.
3. **Calculations:** They are the rules that determine one or several parameters. They are the majority in the present case. Their action part can be constituted by Le–Lisp [14] expressions to be evaluated during the evaluation of their conclusions or can call external softwares.
4. **Cooperations:** They are the rules that establish the necessary connections between the different modules. Their main function is to recuperate the works realized by one of the modules and send them to the others; thus they allow the inference engine to reach an intermediate design result.
5. **Acquisition of data:** They are the rules that initialize the constraints and data of the modules, by means of questions to the user.
6. **Optimizations:** They are the rules that allow the optimization of the criteria previously described. These rules are structured around five major categories according to the limitation of the inverter input DC voltage, the limitation of the frequency, the choice of the inverter command and control, the limitation of losses and the limitation of semi–conductor currents.

This distinction is undertaken with the objective to manage dynamically the tasks tree (Fig. 9).

4.2.3 Rule bases

At each cycle, the inference engine (the program which has to deal with the rules) filters the rules for which

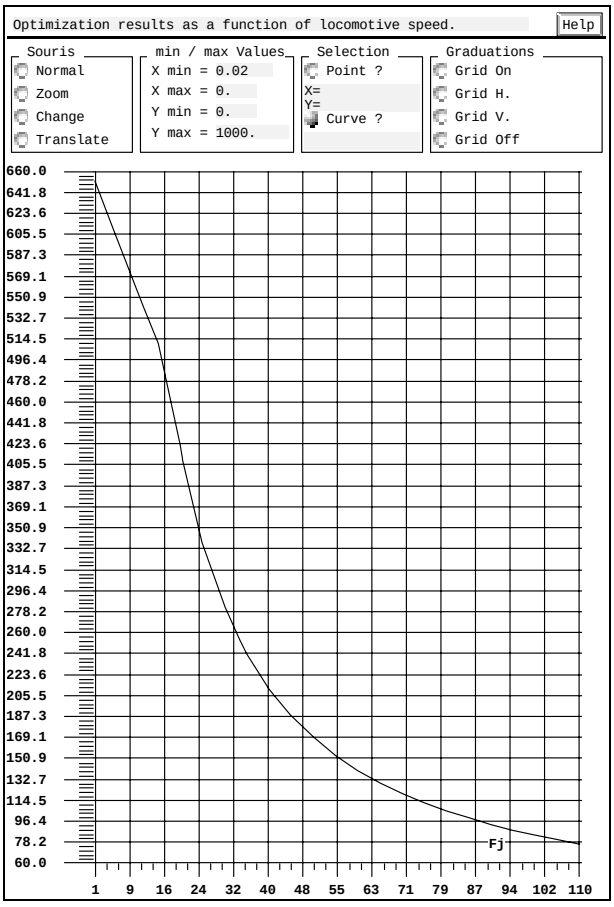

Fig. 11. Specifications: traction effort as function of speed.

hypothesis are verified and then fires them; in order to reduce the risk of investigating useless reasoning, the set of rules is structured in subsets: the rule bases. These rule bases also define the strategy used by the inference engine: for example, it can be interesting, at a given step of the optimization (using a given rule base) to investigate several solutions in parallel (at different initializations of the inverter command for example), or to focus on and investigate completely a reasonings line before considering a new solution. The rule bases are associated with tasks which are used by SMECI to control the operation of the inference engine and can be considered as classical subroutines.

In our system, specialized rule bases are associated with the major subsystems of the railway traction chain: the modules which realize the design of the the *motor*, the *inverter*, the *firing pattern* of the inverter, the *switches* and the *snubbers*.

5 Work session example–validation

The prototype expert system is tested on the concrete case of a railway traction chain represented by the basic diagram of the Figure 1. The specifications are defined by the curve of traction effort as a function of speed (Fig. 11).

5.1 Delivered data

After having started the session, the resolution begins with the creation of the user interface; in this former four zones are displayed (left part of the interface): two zones allow the user to introduce *data* as well as *constraints* for a module; the two other zones present the *informations* and the *output* of the same module and are reserved for the consultation (Fig. 12).

These four zones allow also the visualisation of the progress of the reasoning while *data* are defined by the system.

5.2 Constraints

At the level of constraints, it has been given from the start (Tab. 1):

Table 1. User constraints.

I_{max}	1600 A
U_{max}	1800 V
U_{min}	900 V
P_{max}	2000 W
ΔI_{max}	75 A

It is necessary to note that the maximum peak–to–peak current allowable is 3200 A (two semi–conductors in parallel by inverter leg).

5.3 How to recuperate results

The user has different ways to analyze the results of a session. During the resolution, the results are printed on the terminal (LISP Interaction, Fig. 12). In the "Exchange between modules" window, we can visualize cooperation messages between the different expert modules.

Another way to recuperate results is to obtain explanations on the states using the inference tree generated by the system during the resolution of the problem (in the considered case, all states are numbered and each application of rules corresponds to the creation of a new state). By selecting a state of the tree, the system provides explanations on the concerned state: it gives the name of the rule that comes to be applied, the task that led to this action as well as all object slots modified in this state. For example Figures 13 and 14 show a part of the inferences that led to finding the presented solution.

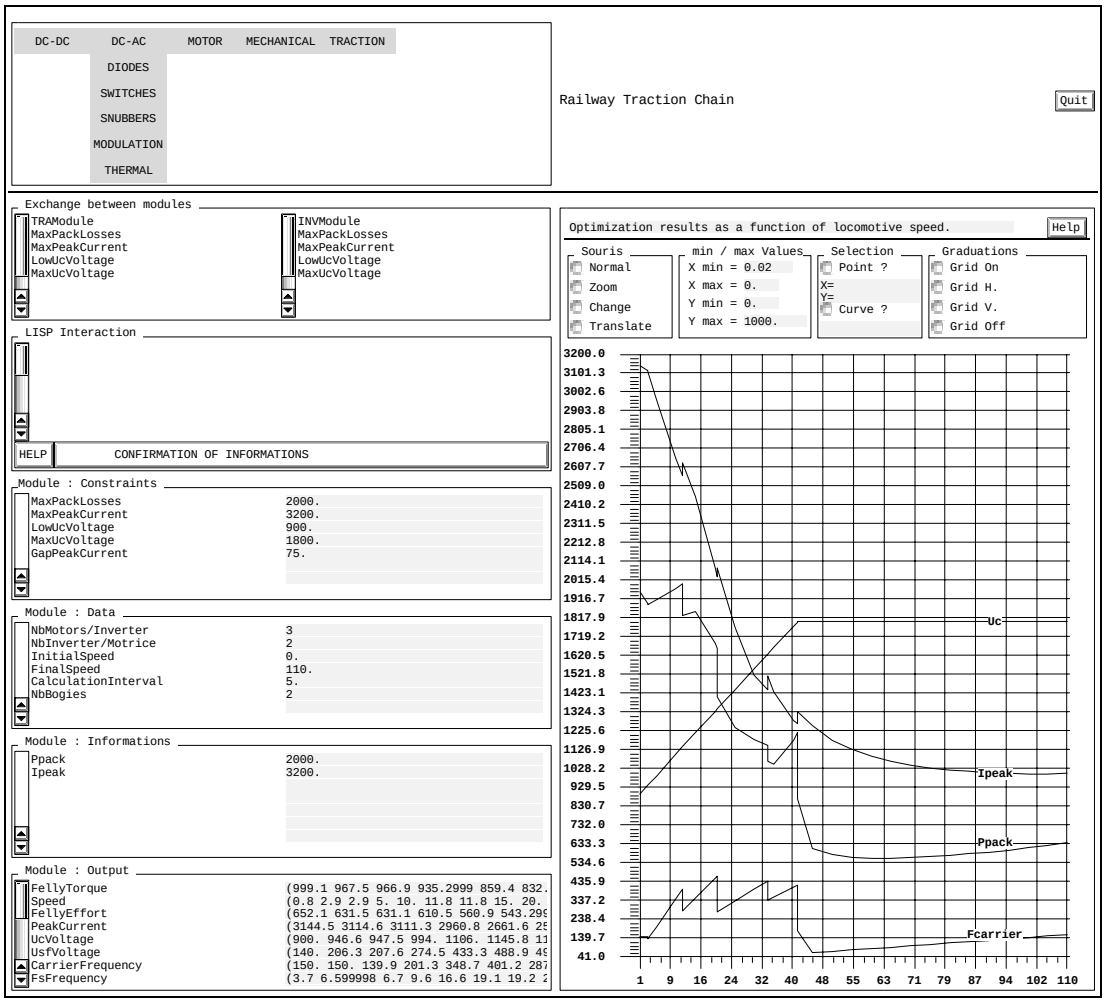

Fig. 12. User interface.

The user can also obtain explanations from objects that retrace the history of modifications to an object slot. It concerns therefore the comments of an applied rule.

In this approach, the exploration of the states is undertaken according to the "Depth First Strategy" which controls the inference cycle, since it defines the status of the created state and imposes the choice of expanded new state for the following cycle. During each cycle, the inference engine chooses to continue the reasoning from the non–singular state.

For the specified constraints (see Tab. 1), the system gives a similar solution to the obtained results of the TGL–ED experts. These results are expressed in the form of curves representing the electrical and mechanical waveforms relative to the component of the circuit (Fig. 1) as function of speed, and whose principals are illustrated by the following figures.

Figure 15 gives the evolution of the motor torques as function of speed for a frequency of 150 Hz at the start. Figure 16 shows the losses, the peak–to–peak current, the inverter DC voltage as well as the inverter Switching frequency. Figure 17 illustrates the evolution of the fundamental motor current and voltage.

6 Conclusion

The proposed study concerns the automatic design of static converters and especially of the inverter–motor of a railway traction chain. The aim is the development of a tool bringing an assistance in the analysis of the specifications, the choice and design of the best structure for the considered application and the optimization of the constraints at the inverter level.

For the development of the prototype, and among the different possible methods in electrical engineering, we have chosen an approach based on artificial intelligence techniques (expert system) as a global approach that offers the possibility to integrate different forms of "know–how" and a great flexibility in its representation. Another advantage of expert systems is that the knowledge base, which implements all the specific information related to the past, can be corrected, updated or extended by a user without the need to resort to computer programming.

The characterization of the components of the considered railway traction chain, according to different criteria and data of the specifications, has led to the development of design autonomous expert modules dedicated to the

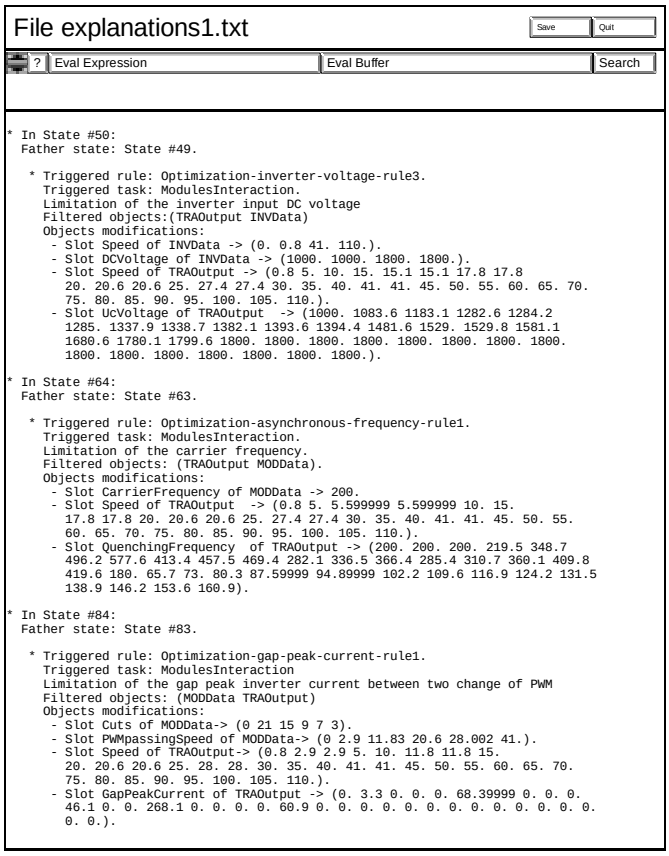

Fig. 13. Explanations of the states 50, 64 and 84.

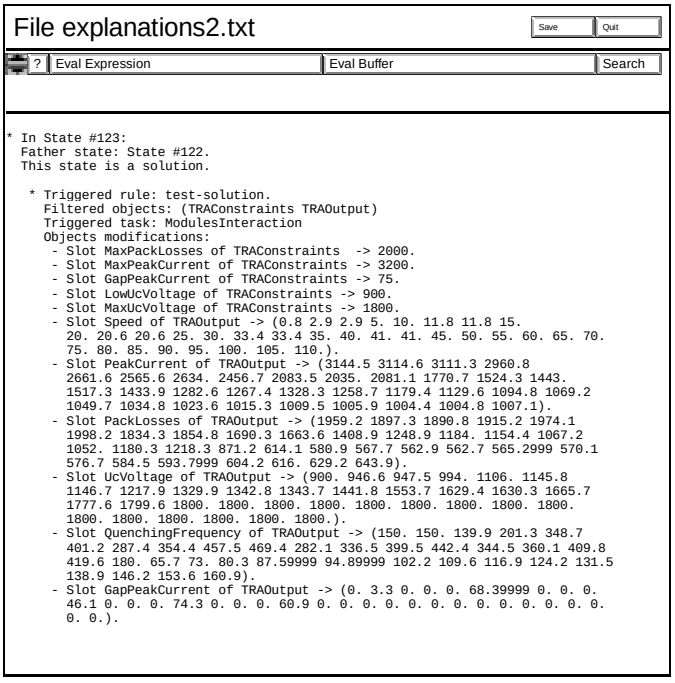

Fig. 14. Explanations of the state 123: this state is a solution.

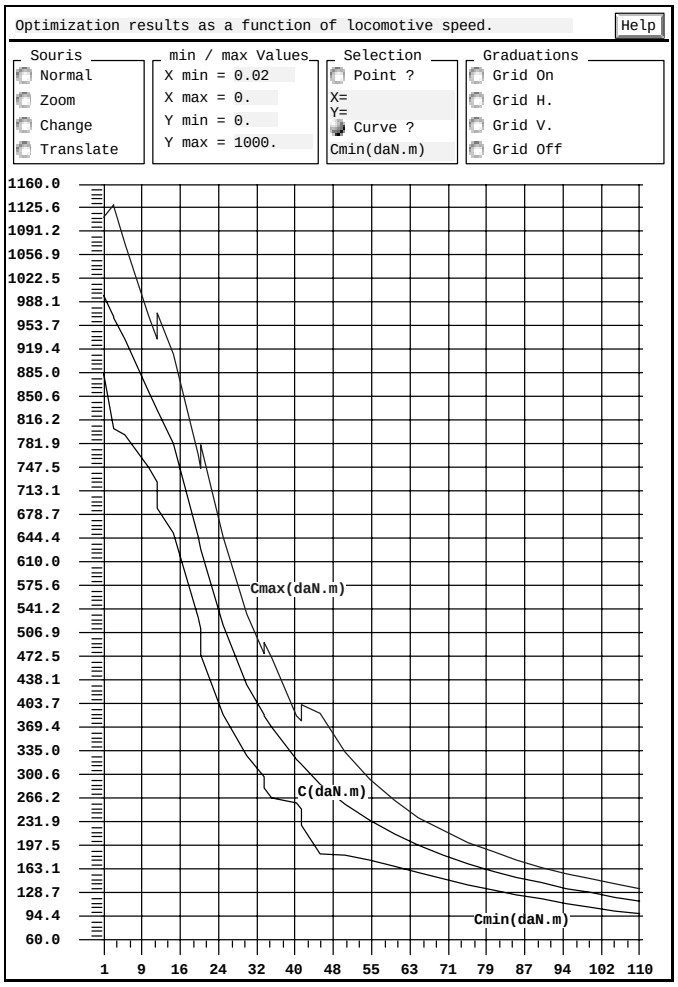

Fig. 15. Evolution of the motor torques.

definition of the different constitutive elements of the railway traction chain (inverter, motor, switches, *etc.*). In order to take into account the modules interaction, it has been necessary to establish a supervisor which allows the cooperation of the different modules implemented and respects their modularity. This supervisor tries to reproduce the process of human designers and constitutes a first step towards the design of static converters using an expert system.

As a design tool, it is able to apply automatically a set of general rules. Its modular structure allows to reuse and to extend the developed modules when another kind of converter is to be studied.

The proposed work concerns the design of a railway traction chain, and we think that this methodology can be used in any AI oriented systems related to the electrical engineering domain.

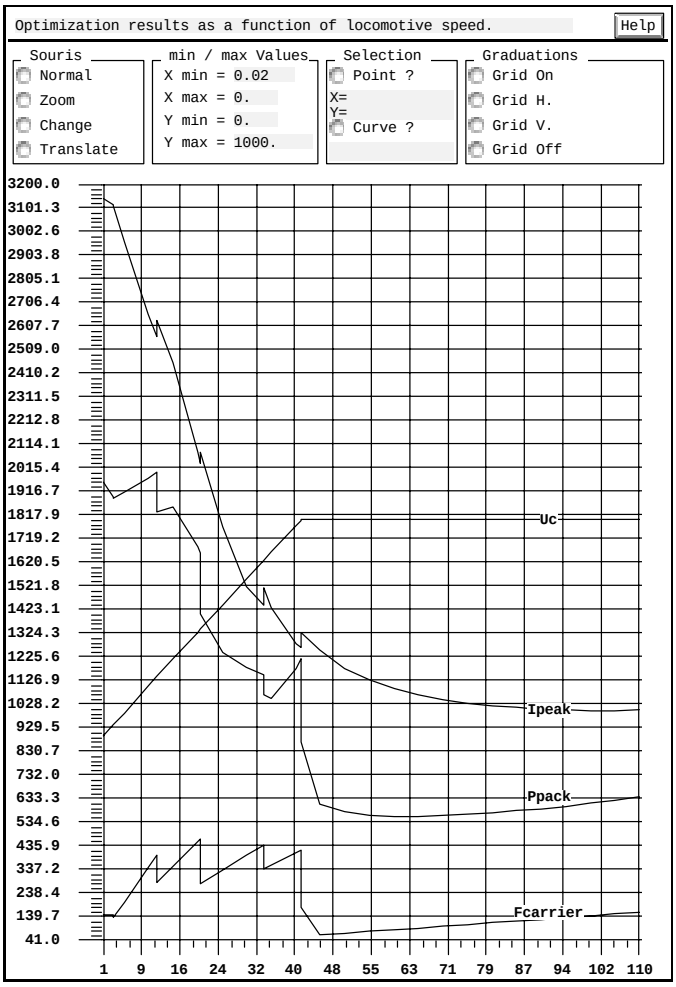

Fig. 16. Evolution of the inverter waveforms, where (U_c) is the DC input voltage, (I_{peak}) is the peak–to–peak current, (P_{pack}) is pack losses and ($F_{carrier}$) is the Switching frequency.

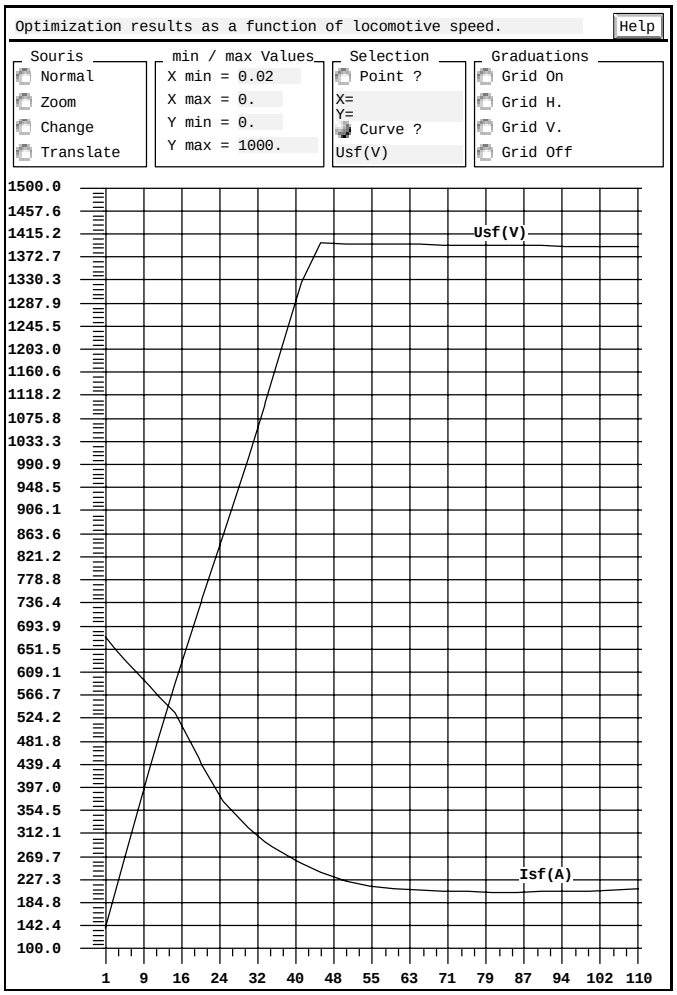

Fig. 17. Optimized results: evolution of the fundamental motor current and voltage.

7 Notations and list of symbols

U_c	V	input DC voltage (U_{min}: lowest, U_{max}: maximum),
F_j	kN	felly effort,
C	daN. m	felly torque (C_{max}: maximum, C_{min}: lowest),
V_{PO}	km/h	full wave pass speed,
$viti$	km/h	locomotive initial speed,
$vitf$	km/h	locomotive final speed,
f	Hz	asynchronous PWM frequency f_{max}: maximum, f_{min}: lowest),
P_{pack}	W	pack losses,
I_{peak}	A	peak–to–peak inverter current,
I_{max}	A	maximum IGBT current,
P_{max}	W	maximum pack losses,
ΔI	A	gap current between two change of PWM ,
U_s	V	interlinked voltage,
I_{sf}	A	fundamental motor current,
U_{sf}	V	fundamental motor voltage,

References

1. D. Chen, B.K. Bose, *Expert System Based Automated Selection of Industrial AC Drive,* in *Proceeding of IEEE IAS*, Houston Texas, (Oct. 1992), pp. 387–392.
2. S.M. Chhaya, B.K. Bose, *Expert System Based Automated Design Technique of a Voltage–Fed Inverter for Induction Motor Drive,* in *Proceeding of IEEE IAS Annual Meeting* (1992).
3. D. Fezzani, *Système expert pour la conception en électro nique de puissance; applications sans interruptions,* Ph. D. thesis, LEEI–INPT, Toulouse, 1996.
4. D. Fezzani, H. Piquet, H. Foch, J. Phys. III France **6**, 349–361 (1996).
5. D. Fezzani, H. Piquet, *EPE Journal*,**6** 72–79, (1996).
6. S. Ahrens, I. Kortaberria, J. Bigeon, D. Poupart, *Expert System for Designing Converters for Large Synchronous Machines,* in *Proc. of the conference EPE*, Firenze, Italy, (Sept. 1991), pp. 266–271.
7. H.B. Puttgen, J.F. Jansen, IEEE Trans. Pow. Syst. **3**, 254–260 (1988).
8. T. Novak, J.R. Meigs, R.L. Sanford, IEEE Trans. Ind. Appl. **25**, 691–698 (1989).

9. K. Debebe, V. Rajagopalan, T.S. Sankar, *Expert Systems for Fault Diagnosis of VSI-FED AC Drives*, in *IEEE IAS conference record, 91-CH-3077-5* (1991), pp. 368–379.
10. S. Ahrens, *Système expert d'aide au diagnostic et à la maintenance des grands entraînements électriques*, Ph.D. Thesis, LEG–INPG, Grenoble, 1993.
11. B. Valiquette, G.L. Torres, D. Mukhedkar, "An Expert System Based Diagnosis and Advisor Tool for Teaching Power System Operation Emergency Control Strategies," IEEE Trans. Pow. Sys. **6**, 1315-1323, (1991).
12. D. Fezzani, H. Piquet, Y. Chéron, and M. Metz, Design of Static Converters: an Expert–System Approach, in *Proc. of the conference EPE*, Brighton (GB), Sept. 1993, pp. 47–52.
13. SMECI, *Manuel de référence - Version 1.76*, ILOG, 2 avenue Galliéni - BP 85 94253 Gentilly - france, 1994.
14. Le-Lisp, *Manuel de référence - Version 15.25*, ILOG S.A., 2 avenue Galliéni - BP 85 94253 Gentilly - france, 1991.
15. J.C. Pomerol, *Les systèmes experts*, (Les Éditions Hermès, Paris, 1988).
16. J.F. Galloüin, *Transfert des connaissances ; systèmes experts : techniques et méthodes*, (Les Éditions Eyrolles, Paris, 1988).
17. C.J. Harris, *Application of Artificial Intelligence to Command & Control Systems*, (Peter Peregrinus Ltd, London 1988).
18. W.J. Black, *Les systèmes intelligents basés sur la connaissance*, (Éditions Masson, Paris, Milan, Barcelone, Mexico, 1988).
19. H. Farreny, M. Ghallab, *Eléments d'intelligence artificielle* (Éditions Hermès, Paris, Londres, Lausanne, 1987).
20. M. Gondran, *Introduction aux systèmes experts* (Éditions Eyrolles, Paris, 1984).
21. J. Laurière, *Intelligence artificielle : résolution de problèmes par l'homme et la machine* (Editions Eyrolles, Paris, 1987).
22. B. Meyer, *Object-Oriented Software Construction* (Prentice Hall International, 1988).
23. B. Meyer, *Conception et programmation par objets pour du logiciel de qualité* (InterEditions, Paris, 1990).