INT. J. PROD. RES., 1998, VOL. 36, NO. 2, 417–436

From CAD to computer aided welding

O. LEGOFF†* and J. Y. HASCOËT‡

Within the context of integrated design, we propose a new approach for off-line programming of welding robots by interfacing a CAD modeller (geometric database) and an artificial intelligence system (welding database). The CAD system associated with our development, used to design the parts to be assembled, allows us to generate welding paths automatically and to extract the assembly features required to determine welding parameters. With these features, we propose a new approach to generate welding parameters automatically in the GMAW process with neural networks. We have chosen to use backpropagation neural networks as this approach integrates database and modelling aspects. Moreover, a neural net based system can easily be improved, it can enlarge its field of application using new experimental welding data. In this paper we present the system we have developed for the generation of paths and then an approach using neural networks to determine welding parameters. We show how CAD features can be used to determine the welding process, the welding wire and then to compute welding parameters.

1. Introduction

Programming by apprenticeship is the mode currently used on almost all continuous welding robots. This kind of programming allows the operator to have an immediate control of actions, but imposes the immobilization of the production tool during apprenticeship. On the other hand, off-line programming is done outside the production site. In this approach, real elements of the welding cell are replaced by computer models. This preparation can be realized with robot programming languages or with graphic supports (computer aided design). The purpose of our work is to ensure the coupling of a CAD modeller, used to design parts to weld, and of an artificial intelligence system which can automatically determine welding parameters in the case of arc welding with the GMAW (Gas Metal Arc Welding) process.

In manual or semi-automated welding, the welder determines the welding parameters (current, tension, welding, speed, etc.) and the paths to follow. CAD-robotic applications currently available allow the operator to define the path point after point by using the models of a CAD system. Nevertheless, they do not exploit the geometrical elements to automatically generate both the paths and the welding parameters. The choice of these parameters has to be proposed by the user, while it could have been deduced from the features of the CAD models. Contrarily, some softwares use the experience linked to the welding trade to generate operating parameters, but weak points of these applications are the CAD aspects and the definition of paths.

Revision received March 1997.

† Laboratoire Universitaire de Recherche en Production Automatisée, École Normale Supérieure de Cachan, 61, avenue du Président Wilson, 94235 Cachan Cedex, France.

‡ Laboratoire d'Automatique de Nantes École Centrale de Nantes, 1, rue de la Noë, 44072 Nantes Cedex-France.

* To whom correspondence should be addressed.

The first part of our work, based on feature extraction, allows the operator to generate welding paths and to extract the information needed for the choice of welding parameters. Then, we propose an approach for the automatic generation of these parameters. We have studied several methods before choosing to use back-propagation neural networks.

In this paper we present the principles used for the generation of welding paths. Then we explain in detail the process we have developed to take into account elements that intervene in the choice of welding parameters.

2. CAD and feature extraction

The first part of our work exploits information contained in CAD models of elements to assemble in order to generate the information needed for robotized welding. We are going to define the components of a welding path before giving the principles of feature extraction and an example application.

2.1. *Definition of a welding path*

Elements that have to be extracted from CAD models are (Fig. 1):

- Joint paths defined in the space by a set of successive positions and orientations of a local frame R_J (Detriche 1987), linked to beads, in relation to a reference frame R_0. These local frames are linked by geometrical elements describing the bead bottoms.
- Characteristics that will allow us to choose welding parameters: angles between parts, thicknesses, material and joint orientation already contained in the path description.

2.2. *Feature extraction*

At this stage, the only data for the problem are the CAD models of the parts. We have chosen to use a parametric surface representation of the objects that allows us to compute all the elements needed for the generation of paths and for the extraction of the assembly features. Our application uses a Bézier model but any other parametric model can be used (B-spline, Nurbs, etc.). A welding path is therefore partly

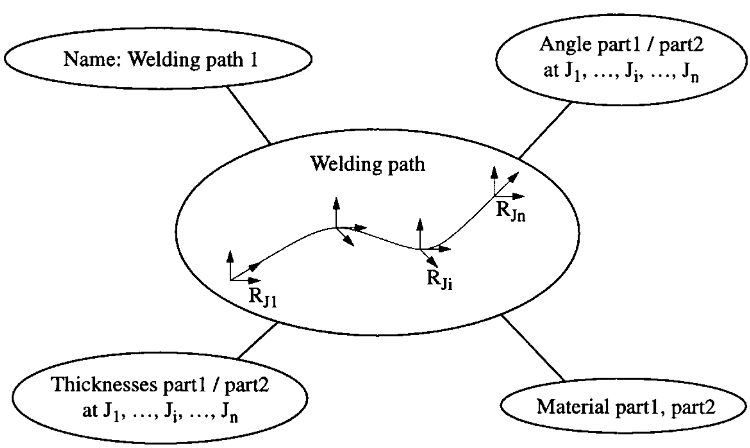

Figure 1. Welding path features.

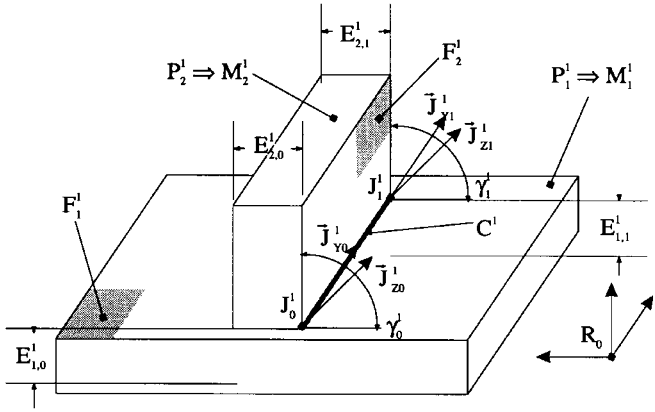

Figure 2. Components of a simple path T^1.

composed of a set of n parametric curves that represent the bottom of the bead, limited by $n + 1$ characteristic points. Considering the initial portion of a path, or a 'simple' path, noted T^1, defined between two parts P_1^1 and P_2^1, the components are (Fig. 2):

- C^1: the parametric curve that describes the bottom of the bead.
- R_{J0}^1 and R_{J1}^1: frames at characteristic points, themselves defined by:
  - J_0^1 and J_1^1: characteristic points,
  - $\overrightarrow{J_{Y0}^1}$ and $\overrightarrow{J_{Y1}^1}$: longitudinal axis that indicates the joint orientation,
  - $\overrightarrow{J_{Z0}^1}$ and $\overrightarrow{J_{Z1}^1}$: normal axis that corresponds to reference orientations for the welding torch.
- γ_0^1 and γ_1^1: angles between parts.
- $E_{1,0}^1, E_{1,1}^1, E_{2,0}^1$ and $E_{2,1}^1$: respectively thicknesses of P_1^1 at J_0^1, of P_1^1 at J_1^1 then of P_2^1 at J_0^1 and of P_2^1 at J_1^1.
- M_1^1 and M_2^1: materials which make up P_1^1 and P_2^1.

The process developed to extract all these data uses classical geometric operations: calculations of normals, tangents, surface intersections, etc. Nevertheless the extraction of certain features, such as thickness, can be difficult according to the element geometries. The different operations are not described here but the details can be found in Legoff (1995). The definition of 'complex' paths, whose beads are composed of several parametric curves, consists in chaining several simple paths. However, torch orientations ($\overrightarrow{J_{Zj}^1}$ vectors) at characteristic points are defined by the bisecting vectors of initial vectors stemming from simple paths. Indeed, during welding along a continuous path, at a given point, it is impossible to wait for a torch reorientation that would produce an undesirable deposit of metal. So a single intermediate orientation has to be used.

2.3. *Application*

During the different stages of the creation of the welding paths the user only has to choose the parts to weld and the paths on which he wants to create a joint. It is not always easy to automate the choice of beads to realize to ensure a correct assembly, for this choice depends on many parameters which are not included in computer models of elements (environment, supported efforts, etc.). The proposed methods have been developed within a CAD modeller (Euclid). Figures 3 to 6 show path examples defined for the welding of a motorbike framework. The system shows path representations and torch orientations; the characteristics that are not represented (thickness, angles, etc.) are indicated to the operator and kept for the determination of welding parameters.

The approach we propose allows us to extract all the characteristics of the assembly that will be used for the welding robot program. Paths and torch orientations can be used to simulate and to program the robot movements; the other features, needed for the choice of welding parameters, are exploited later.

3. Neural networks and welding parameters

Because of the lack of knowledge of the phenomena that intervene in the welding electrical arc, there are no theoretical models for choosing welding parameters. An automatic choice of these parameters therefore has to use the experience and the know-how which exist in this field (experimentation, knowledge of experts). The computer exploitation of this knowledge uses various approaches such as expert

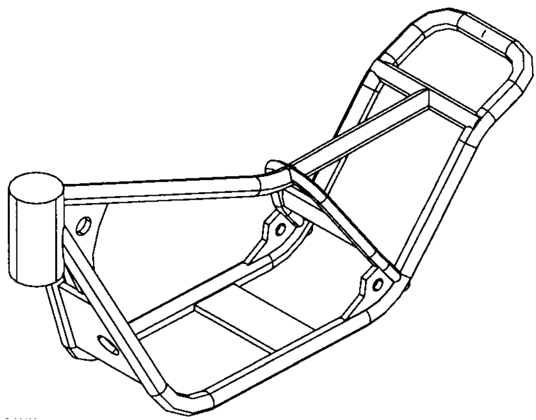

Figure 3. Motorbike framework.

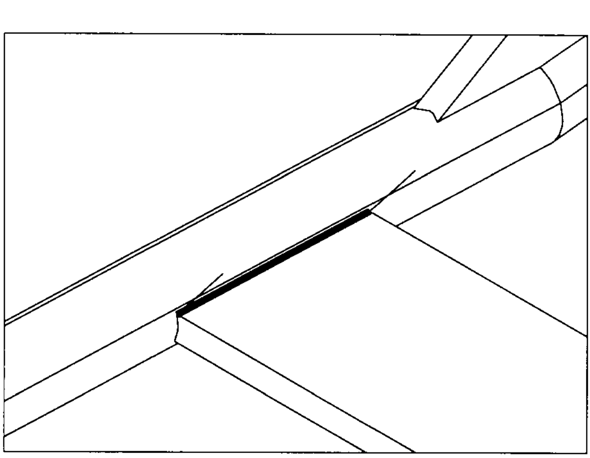

Figure 4. Plane/cylinder welding.

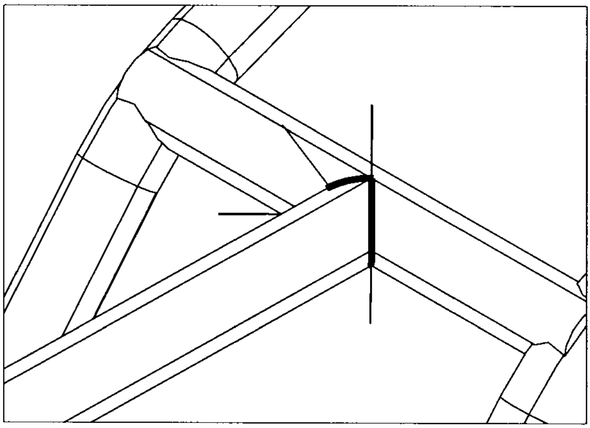

Figure 5. Cylinder/cylinder welding.

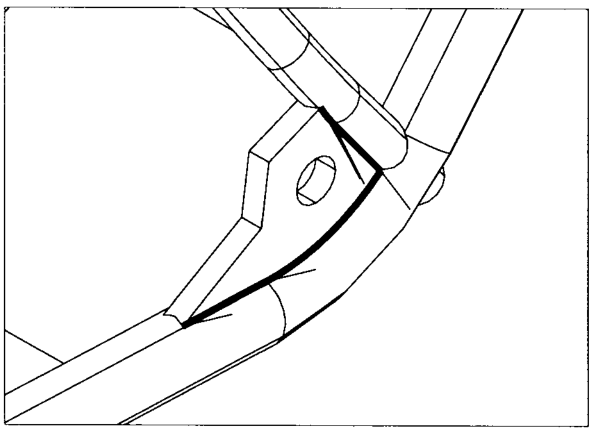

Figure 6. Welding of 3 parts.

systems database or the data analysis. Our objective is to propose an artificial intelligence system which can provide welding parameters for a given assembly. Considering the diversity of welding cases that can appear, it is essential for this system to be evolutionary and auto-adaptative to the user's needs, so it must be able to take into account new welding configurations (processes, welding positions, welded materials, etc.). We have studied several methods, commonly used in welding, before turning our work on to a new approach: neural networks, which are seldom used in this area and more generally in the field of CAD and of integrated design.

3.1. *Studied methods*

The methods used (Barborak *et al.* 1991, Bernasek 1991, Breat *et al.* 1992) try to take into account the experience linked to the welding trade with different data-processing techniques.

3.1.1. *Expert systems*

These systems are well adapted to diagnose and propose modifications of adjustments when a welding test has not given the expected results. The MIG-expert product, developed by the French Welding Institute, uses such a system (Breat and Pauwels 1991, Breat *et al.* 1992). However, the functioning principle of the expert system, based on the exploitation of type *If* ... *Then* ... rules, does not seem very appropriate for reaching our objective. First, the determination of welding parameters is more similar to database or modelling aspects, moreover updating an expert system requires writing new rules, always a delicate stage where a programmer has to translate the principles exposed by the expert into rules understandable to the system. So, after having studied expert system possibilities and envisaged the diversity and the quantity of rules that would be needed to generate welding parameters automatically, we have preferred to turn our work towards other methods.

3.1.2. *Data analysis and modelling*

Modelling from experimental examples is one of the methods often used in welding (McGlone 1979, Alberry 1989, Galopin 1989). The aim is to find a model of the studied phenomenon. The models frequently used are stemming data analysis. The initial model is linear and a least-square approximation provides the model parameters that minimize this criterion. From this method we have tested linear, logarithmic, exponential, and power models (Legoff 1995). Our works in this field have not given satisfactory results. The necessity of a great number of tests and the impossibility of taking into account qualitative variables strongly limit the interest of this approach, particularly in our case where we need to be able to update the system easily.

3.1.3. *Database and interpolation*

The use of a database is the simplest solution and it presents the advantage that it can be very easily updated. However, such a system can only repeat the welding cases that it has learnt. To remedy this problem we can search for the closest case to the one to be dealt with, if this case is not known by the database, or several close cases in order to interpolate. It is the approach proposed by the French Welding Institute (Breat and Pauwels 1991, Breat *et al.* 1992). This is a very interesting idea but the application is delicate. The first difficulty is the choice of a measure of resemblance between cases to deal with process and known data. Criteria that we have used are (Diday *et al.* 1982): the Euclidian norm, the χ^2 distance, the absolute value and the Cambera distance. Despite difficulties linked to this method (Legoff 1995) results are interesting and updating the initial database is very simple. Nevertheless the neural approach permits one to obtain better results for the same welding database.

3.1.4. *Neural networks*

After having tested several approaches to building our system, we have studied a new discipline in full evolution: neural networks. There exists some neural network applications to welding concerning the bead section modelling according to welding parameters (Andersen *et al.* 1991, Sutter and Xu 1993), and the on-line control of weld bead (Anderson *et al.* 1990). One of the main points of neural network approaches is that they are based on learning from examples, which is particularly our problem. The characteristics of a problem well adapted to solving by neural networks (Davalo and Nain 1990):

(1) The rules needed to solve the problem are unknown or really difficult to explain and formalize. However, experts are able to propose a set of examples corresponding to the problem inputs and outputs (solutions).
(2) The problem uses noisy data.
(3) The problem can be evolutive (taking into account new data).
(4) The problem requires a high speed of processing.
(5) There is no technological solution.

We notice that our problem can be solved with neural networks. One of the main points is the possibility of evolution that allows us to enlarge the initial field of application by considering new reference examples. Furthermore, this approach has given the best results on a test database, so this is why we have developed our system with neural nets.

3.2. *Backpropagation neural networks*

We propose a description of the neural networks we have used. Today the term neural networks gathers many models which tend to mimic some functions of the human brain by reproducing some of its structures. The first formal neuron model was presented by McCulloch and Pitts in the forties (McCulloch 1943). Here, we detail neither the biological foundations of these models, nor all the architectures and kinds of neural networks: they can be found in the literature (Lippman 1987, Davalo and Nain 1990, Bourret 1991). For all neural networks we distinguish a learning stage and a using stage. In our application we use multi-layered feed-forward networks which are the most used (in 90% of practical applications).

3.2.1. *Model of a multi-layered neural network*

During the use stage, such a system acts as a 'black box' with inputs and outputs. The links between layers are the lively elements. During the learning stage an algorithm teaches the system the values to give these links. The backpropagation algorithm has been developed by Rumelhart *et al.* (1986); the typical structure of multi-layered neural net is represented in Fig. 7. Each neuron is connected to all neurons of the following layer by links whose weights $w_{i,j,k}$ are real numbers. Each

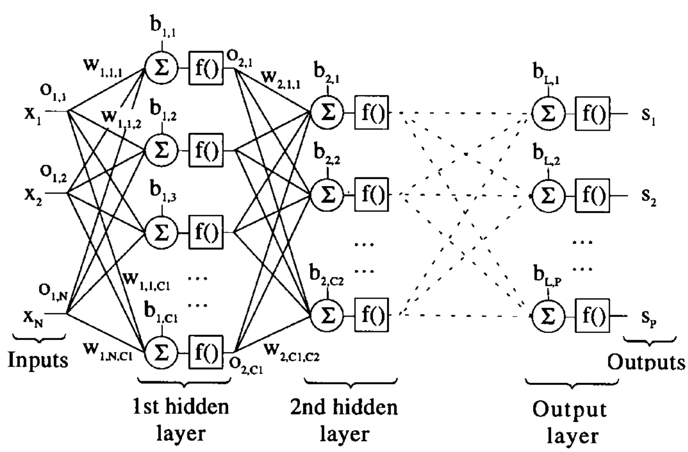

Figure 7. Model of a multi-layered neural network.

coefficient $w_{i,j,k}$ is the weight of the link from the neuron j of the layer $i-1$ towards the neuron k of the layer i. System variables are the following:

$X = (x_1, x_2, \ldots, x_N)$ is the input vector,
$Y = (y_1, y_2, \ldots y_P)$ is the reference output vector,
$S = (s_1, s_2, \ldots, s_P)$ is the computed output vector of the system,
$w_{i,j,k}$ are the weights affected to each link between the layers $i-1$ and i,
$b_{i,k}$ are bias associated to each neuron of the layer i.

The structure of the system is defined:

- The number of layers composing the net (minimum: one output layer).
- The number of neurons in each layer (the last layer is composed of a number of neurons P equal to the number of outputs).
- The activation functions f used for the neurons (they can be different for each neuron, but are the most often identical for a layer and for the whole system, that is the case in Fig. 7). Generally the logarithmic sigmoid function is used (although other functions can be used):
$$f(x) = \frac{1}{1+e^{-x}}$$
- The entry of f, noted $I_{i,k}$, is computed for the neuron number k of the layer i by:
$$I_{i,k} = \sum_j w_{i,j,k} o_{i,j} + b_{i,k}$$
  where $o_{i,j}$ is the output value coming from the layer $i-1$ by the link $w_{i,j,k}$. So:
$$f(I_{i,k}) = o_{i+1,k} \quad \text{and} \quad f(I_{L,k}) = s_k \quad \text{for the output layer.}$$

3.2.2. *Backpropagation learning*

The aim of the learning stage is to determine the values of the coefficients $w_{i,j,k}$ and $b_{i,k}$ by minimizing an error-function. It is achieved by presenting to the system learning examples after a random initialization of weights and bias several times if necessary. The error-function to minimize is the sum-squared error between computed outputs and real outputs. Once the partial derivative for each weight and bias is known, the aim of minimizing the error-function is achieved by performing a simple gradient descent. The error-function for all the M examples is defined by:

$$E(w,b) = \sum_{m=1}^{M} E^m(w,b)$$

with $E_m(w,b)$ the error on the example m:

$$E^m(w,b) = \sum_{p=1}^{P} \left(s_p^m - y_p^m\right)^2$$

So, the gradient descent gives the weights and bias modification rule:

$$w_{i,j,k}(q) = w_{i,j,k}(q-1) - e(q)\frac{\partial E}{\partial w_{i,j,k}}$$

where $e(q)$ is the gradient step or 'learning rate' at epoch q.

After calculations and taking into account the use of the sigmoid function, we obtain

$$w_{i,j,k}(q) = w_{i,j,k}(q-1) - e(q)d_{i,k}o_{i,j}$$

where

$$d_{L,k} = 2\sum_{m=1}^{M}(s_k^m - y_k^m)s_k^m(1 - s_k^m) \text{ for the output layer,}$$

$$d_{i,k} = \sum_{h} d_{i+1,h}w_{i+1,k,h}o_{i+1,k}\left(1 - o_{i+1,k}\right) \text{ for hidden layers,}$$

where h focuses on the neurons of the next layer, for $i = L-1$, we have $h = 1,\ldots,P$ on the output layer L.

Bias $b_{i,k}$ are learnt as other weights assuming that it concerns a constant entry value equal to 1. The learning stage is stopped if the sum-squared error $E(w,b)$ is lower than an error goal, i.e. $E(w,b) \leq E_0$, or if a maximum number of epochs Q has been reached.

3.2.3. *Limits and improvements of the algorithm*

The use of the backpropagation algorithm for neural networks learning gives good results in many applications. However, some difficulties remain:

- Initialization of link weights greatly influences the algorithm, notably the convergence point of the network.
- The convergence speed is not controlled.
- There is no design method for a system (number of layers, number of neurons in each layer, etc.) to solve a given problem.
- Progression values and stopping criteria of the algorithm (gradient step, maximum number of epochs, error goal) have to be determined by practical experience.
- Data have to be pre-processed to ensure the algorithm convergence. This pre-processing is generally a reduction of inputs and outputs to a similar scale (generally from 0 to 1, −1 to 1 or −0·5 to 0·5).

Much research work aims to improve weak points of the method, and it is very unusual to use the backpropagation algorithm, such as has been presented, without modifications. Classical improvements of the backpropagation are:

- The use of an adaptive learning rate (Demuth and Beale 1991, Vogl *et al.* 1988), where $e(q)$ is modified according to the evolution $E(w,b)$ during learning;
- The introduction of a ‘momentum term’ in the weight modification rule (Lippman 1987, Vogl *et al.* 1988, Bourret *et al.* 1991, Demuth and Beale 1991), which limits weight oscillations during learning. The modification rule then becomes:

  $$w_{i,j,k}(q) = w_{i,j,k}(q-1) - e(q)d_{i,j}o_{i,j} + \mu\left(w_{i,j,k}(q-1) - w_{i,j,k}(q-2)\right)$$

  where μ is the momentum term $0 < \mu < 1$;
- The use of a new activation function (Harvas Martinez *et al.* 1992, Scalero and Tepedelenlioghu 1992);

- A non-random choice of initial weight values (Nguyen and Widrow 1990).

3.2.4. *Chosen algorithm*

After having used some of these methods of improvement of the initial back-propagation algorithm (momentum term, adaptive learning rate and an initial value choice of weights by the method of Nguyen and Widrow) that have allowed us to obtain interesting results (Hascoët 1994, Legoff 1994a, b), we have chosen to use a new algorithm which is faster than these methods during learning stage: the 'Resilient Propagation' (RPROP) algorithm. This algorithm, proposed by Riedmiller and Braun (1993), is in fact an adaptive learning rate where the gradient step is locally defined for each weight of the network. The modification of weights and the learning rate take into consideration the influences of the considered weight on the error-function. Each weight possesses an update-value $\Delta_{i,j,k}$. During the learning stage this value is modified according to its influence on the error $E(w, b)$. We will not describe here the functioning of RPROP, but another advantage of this algorithm is the reduced number of evolution parameters that has to be *a priori* fixed. It is indeed a problem of classical methods that they generally require several tests before obtaining a set of parameters ensuring a stable evolution of the algorithm and an optimal convergence.

So the parameters used by RPROP are:

- Δ_0: initial values of update-values, so Δ_0 directly determines the size of the first weight-step,
- η^+ and η^-: increase and decrease factors of $\Delta_{i,j,k}$,
- Δ_{min} and Δ_{max}: lower and upper limits of update-values.

It appears that default values of these parameters proposed by Riedmiller and Braun ($\Delta_0 = 0{\cdot}1$, $\eta^+ = 1{\cdot}2$, $\eta^- = 0{\cdot}5$, $\Delta_{min} = 10^{-6}$ and $\Delta_{max} = 1{\cdot}0$) have always given us better results than the classical methods.

3.3. *Design of the system of welding process parameter choice*

Welding processes with which we are connected are gas metal arc welding processes (GMAW) which use a fuse electrode under gaseous protection and this for conventional assemblies. Welding parameters associated to this type of welding processes can be divided into two classes (Cornu 1985): predetermined variables (welding process, electrode, etc.) and operating parameters (current-intensity, arc voltage, welding speed, etc.). These data are of two types: qualitative, in which case they are represented by binary values; or quantitative, when they are centred and reduced.

Initial data of the problem, extracted from the CAD system, are:

- The material that composes the parts, coded as follows:
  $(m_1, m_2) = (1, 0) \Leftrightarrow$ steel,
  $(m_1, m_2) = (0, 1) \Leftrightarrow$ other material.
- The thicknesses E of parts at considered points.
- Three variables that allow us to define the orientation of the joint to weld (Fig. 8):
  α: the rotation angle $0^\circ \leq \alpha \leq 180^\circ$,
  β: the inclination angle $-90^\circ \leq \alpha \leq 90^\circ$,
  γ: the angle between the two parts to joint.

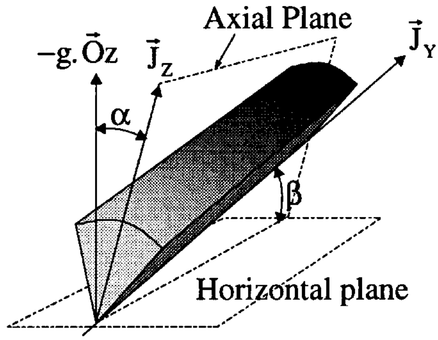

Figure 8. Definitions of α and β.

The system acts as a black box. However, the determination of unknown parameters, predetermined variables as well as operating parameters, has to be done logically. So it is important to structure the order in which parameters are determined. That allows us to reduce the neural network size (number of layers and neurons) and also authorize several entry points. Suppose, for example, that the user has already made a choice of a welding process that he wishes to use. It is then possible to go directly to the next stages taking into account the chosen process for the determination of other elements.

Welding parameters, outputs of the system, are determined in four stages (Fig. 9):

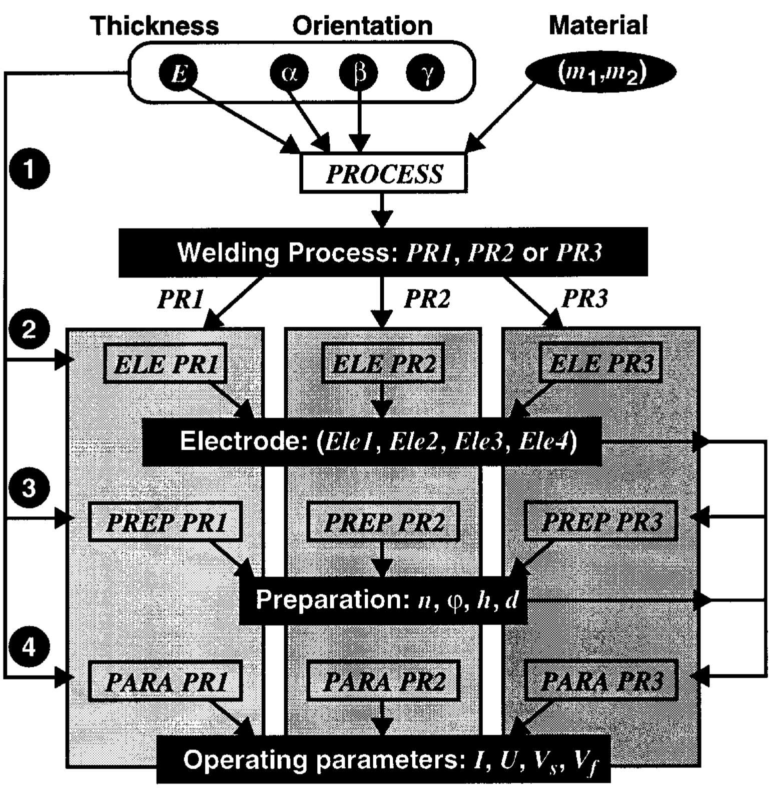

Figure 9. System structure.

(1) A neural network determines the welding process most adapted to the welding problem. In our application three processes, differentiated by their protection gas, are considered: $PR1(Ar + CO_2)$, $PR2(CO_2)$ and $PR3(O_2 + Ar)$. The reference database is composed of 0 and 1, thus when it is possible to use the first two processes we have: $(PR1, PR2, PR3) = (1, 1, 0)$. During the utilization stage of the system, the closer to 1 a response is, the more it is recommended to use the considered process.
(2) Three networks are then used, one for each welding process, in order to choose the electrode. In the same way as for welding processes, electrodes are considered as qualitative information, which allows us to take electrodes of different composition and diameter into account. The four electrodes considered in our application only differ by their diameter:

$$Ele1 \rightarrow \phi = 0{\cdot}8\,\mathrm{mm},\ Ele2 \rightarrow \phi = 1\,\mathrm{mm},\ Ele3 \rightarrow \phi = 1{\cdot}2\,\mathrm{mm},$$
$$Ele4 \rightarrow \phi = 1{\cdot}6\,\mathrm{mm}.$$

(3) The system then determines the number of passes n and preparations to do on parts before welding (Fig. 10): the gap between parts d (mm), the angle of the chamfer φ (degrees) and the height under chamfer h (mm) if it is necessary to make a chamfer.
(4) Finally, the operating parameters are determined. Using a continuous current: the current-intensity I(A), the voltage U(V), the welding speed (the feedrate of the welding torch) V_s (mm/mn) and the welding wire (electrode) melting speed V_f (m/mn).

3.4. *Learning*

We are going to detail the results obtained during the learning stages for each network. As designers of these neural networks, we have had to define for each network: the number of layers, the number of neurons in each of these layers and the two stop criteria E_0 and Q.

Generally speaking, the choice of the structure of neural networks is a delicate problem. It is very difficult to establish general rules about network design. We can be tempted to choose a large number of neurons and layers (more variables to optimize), but it increases the possibilities for the learning algorithm to reach a local minimum and to generate unstable results in the using stage. We can see that in practical applications it is very rare to use a network with more than two hidden layers. So the aim is to choose the minimum number of neurons and layers that allows us to solve our problem. Several papers (Harvas Martinez *et al.* 1992) can guide an initial choice, but experimentation is the only way to validate and definitively choose a neural network structure.

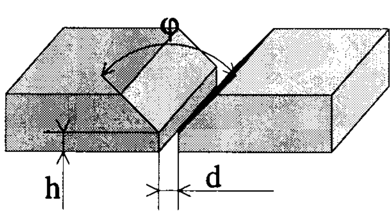

Figure 10. Part preparation.

For our application, we first tested the networks' performances on several typical cases (classification and approximation problems), then we used the following strategy. First, a network has at least one hidden layer with a number of neurons equal to the number of inputs and one output layer. If the learning algorithm converges rapidly (a hundred of epochs) we reduce the number of neurons in the hidden layer. If it converges very slowly, we consider that the initial model is not rich enough, and we increase the number of neurons in the hidden layer before adding a second hidden layer.

As for the choice of stop criteria during the learning phase, the stopping parameter Q is only a limit for the algorithm and its value is arbitrarily chosen. The second parameter E_0 is the sum-squared error that we allow during the learning phase for all output variables. Considering the values these data can take (between 0 and 1) the choice of $E_0 = 0{\cdot}01$ has seemed judicious since it corresponds to a magnitude order of 10% ($E_0 = 0{\cdot}1^2$) of the maximum value of a single variable.

The number of reference examples used for the learning of a network is noted M, at the end of learning the final number of epochs is noted Q_f and the sum-squared error reached is noted E_f. All learning times T_T required for a learning stage have been obtained on a PC 486DX33 and Matlab.

3.4.1. *Welding process: PROCESS*

The network PROCESS allows us to determine the welding process adapted to the considered welding case.

- The structure of the network is the following:
  5 inputs: $E, \alpha, \beta, (m_1, m_2)$,
  3 outputs: $PR1, PR2, PR3$,
  3 layers with 10 + 10 + 3 neurons.
- The learning parameters are: $Q = 5000$ and $E_0 = 0{\cdot}01$.
- The number of examples used and the results are given in Table 1.

We notice that the learning stage gives a precise result with $E_f = 0{\cdot}01$. If we consider the nature of outputs (binary values) the objective is to obtain responses close to 1 or 0 that have been learnt by the network. The final error E_f therefore allows us to assert that all examples taught to the network will be repeated with no significant error during the utilization stage. In order to represent areas of process utilization, we have chosen to express responses of the network for a given material and a given β, variables being thickness and angle α. Figure 11 represents a section of the resulting surfaces if we choose 0·8 as threshold for a favourable response. For each welding process, contours delimit the zones where network responses are superior to this value and correspond to utilization areas.

Network	M	Q_f	E_f	T_T
PROCESS	220	1041	0·01	15′

Table 1. Learning of *PROCESS* networks.

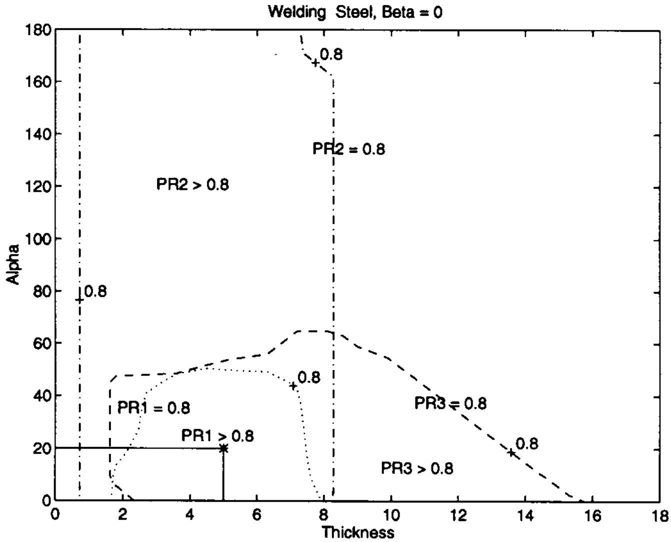

Figure 11. Welding of steel, $\beta = 0$.

3.4.2. *Electrode: ELE*

Knowing the initial data and the welding process to use, three networks are used (one for each process) $ELE\,PR1$, $ELE\,PR2$ and $ELE\,PR3$, to choose an electrode compatible with the joint to weld.

- The structure of the networks is the following:
  4 inputs: E, α, β, γ,
  4 outputs: $Ele1, Ele2, Ele3, Ele4$,
  2 layers: 5 + 4 neurons.
- The learning parameters are: $Q = 5000$ and $E_0 = 0{\cdot}01$.
- The results obtained are given in Table 2.

The precision of the results is satisfactory, it is similar that for networks PROCESS with $E_f = 0{\cdot}01$ for a same nature of outputs (binary). The maximum errors are 0·05 for $ELE\,PR1$, 0·05 for $ELE\,PR2$ and 0·06 for $ELE\,PR3$. We can represent areas of electrode utilizations according to thickness for a determined welding position (Fig. 12). For example the welding with the process $PR3$ of two steel plates with 3·5 mm thickness, $\alpha = 45^\circ$, $\beta = 0^\circ$, $\gamma = 90^\circ$ will be carried out with an electrode of 1 mm diameter.

Network	M	Q_f	E_f	T_T
$ELE\,PR1$	24	102	0·01	17″
$ELE\,PR2$	23	117	0·01	18″
$ELE\,PR3$	34	452	0·01	66″

Table 2. Learning of *ELE* networks.

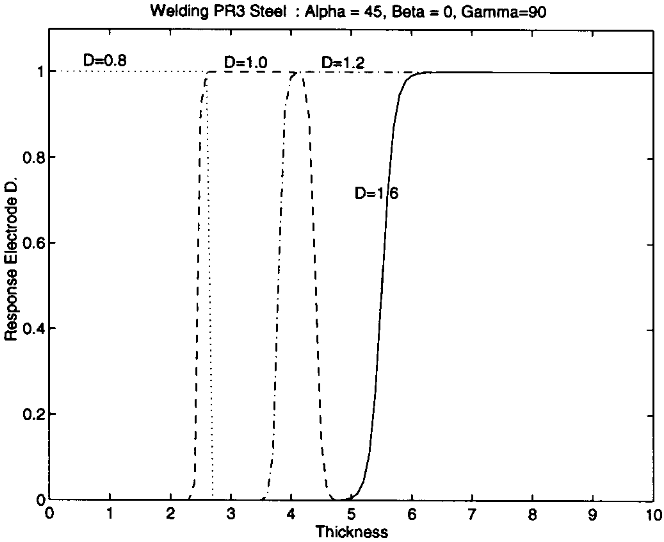

Figure 12. Response of network *ELE PR*3.

3.4.3. *Part preparation: PREP*

In the same way as for the preceding stage, we have three networks with identical structure to determine preparations to undertake on the parts before welding and the number of passes to realize. The three networks are *PREP PR*1, *PREP PR*2 and *PREP PR*3.

- The structure of the networks is the following:
  8 inputs: $E, \alpha, \beta, \gamma, Ele1, Ele2, Ele3, Ele4$,
  4 outputs: φ, d, n, h,
  2 layers: 8 + 4 neurons.
- The learning parameters are $Q = 5000$ and $E_0 = 0{\cdot}01$.
- The results obtained are given in Table 3.

We notice that for process *PR*3 the criterion E_0 has not been reached, nevertheless, as Table 4 shows, examples of learning are reproduced with a satisfactory error edge.

Network	M	Q_f	E_f	T_T
*PREP PR*1	40	2667	0·01	7′10″
*PREP PR*2	36	4336	0·01	0′18″
*PREP PR*3	35	5000	0·05	7′01″

Table 3. Learning of *PREP* networks.

	φ	d	n	h
MAX	60	3	5	1·5
ϵ_{MAX}	2·6	0·13	0·49	0·07

Table 4. Errors obtained.

Network	M	Q_f	E_f	T_T
$PARAPR1$	40	1941	001	20′10″
$PARAPR2$	36	1562	0·01	16′18″
$PARAPR3$	68	13 868	0·01	2h 46′

Table 5. Learning of $PARA$ networks.

	V_s	V_f	I	U
MAX	11	730	420	30
ϵ_{MAX}	0·23	20	9·5	0·39

Table 6. Errors obtained.

3.4.4. *Operating parameters: PARA*

This last stage generates the operating parameters to be used. A new variable is introduced here, it concerns the number of the pass realized $n1$, with $1 \leq n1 \leq n$.

- The three networks $PAPAPR1$, $PARAPR2$ and $PARAPR3$ have the same structure:
  13 inputs: $E, \alpha, \beta, \gamma, Ele1, Ele2, Ele3, Ele4, \varphi, d, n, h, n1$,
  4 outputs: V_s, V_f, I, U,
  2 layers: 20 + 4 neurons.
- The learning parameters are: $Q = 5000$ and $E_0 = 0{\cdot}01$.
- The results obtained are given in Table 5.

During the learning of the network $PARAPR3$ with $Q = 5000$ epochs, the criterion E_0 has not been reached. A new learning with a limit of $Q = 30\,000$ epochs has allowed us to improve the results for this network especially. We observe in Table 6 that the errors made on learning examples are acceptable.

3.5. *Application*

The different networks that compose the system have been trained, it is then possible to exploit them. We present here an example of application but other examples can be found in Legoff (1995). The example we have chosen to treat is represented in Fig. 13, it concerns the welding of a complex joint compound of a vertical descending path portion and an horizonal portion. Considering that welding parameters are constant along a joint portion, from point J_0^1 to point J_1^1 then from point J_0^2 to J_1^2 for our example, we use vectors

$$\overrightarrow{J_{Z1}^1} \text{ and } \overrightarrow{J_{Z0}^2}$$

to define angles rather than the torch orientation represented by

$$\overrightarrow{J_Z^{12}}.$$

So the joint features are:

- Thicknesses of parts: $E = 3\,\text{mm}$.

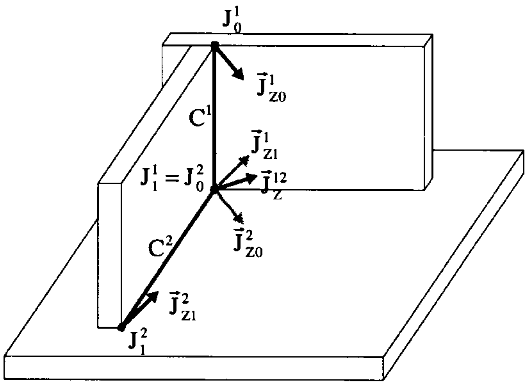

Figure 13. Example of a complex joint.

- Material: steel $(m_1, m_2) = (1, 0)$.
- Angle values:
  - Portion C^1: $\alpha = 90^\circ, \beta = -90^\circ, \gamma = 90^\circ$.
  - Portion C^2: $\alpha = 45^\circ, \beta = 0^\circ, \gamma = 90^\circ$.

The first stage consists in determining the process that has to be used to weld this joint. Responses of the network PROCESS are:

$$C^1: \quad PR1 = 0 \quad PR2 = 1 \quad PR3 = 1$$
$$C^2: \quad PR1 = 0{\cdot}83 \quad PR2 = 1 \quad PR3 = 1$$

So in order to weld all the joints with the same gas it is possible to use the protection gas $PR2$, or Argomix $PR3$. We are going to develop the two possibilities. So for the choice of an electrode:

$$ELE\,PR2 \quad C^1: \quad (Ele1, Ele2, Ele3, Ele4) = (0, 1, 1, 0)$$
$$C^2: \quad (Ele1, Ele2, Ele3, Ele4) = (0, 1, 1, 0)$$

$$ELE\,PR3 \quad C^1: \quad (Ele1, Ele2, Ele3, Ele4) = (0, 0, 0{\cdot}9, 0)$$
$$C^2: \quad (Ele1, Ele2, Ele3, Ele4) = (0, 1, 0, 0)$$

Only the process $PR2$ allows us to use the same electrode for all of the joints to weld, so, it is the solution that has to be chosen in this case. If no welding process had allowed us to obtain a solution with one electrode for all the joints, it would have then been necessary to choose another way to weld these parts, for example to divide this joint into two distinct joints. For the considered example, we have arbitrarily chosen to use the electrode $Ele3$ ($\phi = 1{\cdot}2$ mm) but it is possible to use $Ele2$ or to develop the two solutions.

Then we can determine, with $PREP\,PR2$, the part preparations:

$$C^1: \quad \varphi = 0^\circ \quad d = 0\,\text{mm} \quad n = 1 \quad h = 0\,\text{mm}$$
$$C^2: \quad \varphi = 0^\circ \quad d = 0\,\text{mm} \quad n = 1 \quad h = 0\,\text{mm}$$

These results indicate that parts have to be welded in a single pass ($n = 1$) with neither chamfer ($\varphi = 0^\circ$) nor gap ($d = 0$ mm). Finally, we can determine the operating parameters with $PARA\,PR2$:

$$C^1: \quad V_s = 271\,\text{mm/mn} \quad V_f = 3\,\text{m/mn} \quad I = 130\,\text{A} \quad U = 19{\cdot}7\,\text{V}$$

$$C^2: \quad V_s = 470\,\text{mm/mn} \quad V_f = 5{\cdot}2\,\text{m/mn} \quad I = 151\,\text{A} \quad U = 21{\cdot}7\,\text{V}$$

These paramters have been used to weld the joint with a welding robot and a protection gas close to $PR2$ (Téral 12-Airliquide). The result is entirely satisfactory, no modification of the parameters proposed by our system having been necessary.

4. Conclusions

In order to increase productivity in the area of robotized welding, off-line programming is an essential factor whose objective is to reduce the immobilization duration of the production tool while facilitating the task of the operator. The objective of the work presented in this paper is to automate the off-line programming of welding robots, notably for continuous welding processes with fuse electrode under gaseous protection. In the current context of integrated design and concurrent engineering, our objective is therefore to group three worlds: the world of CAD, the world of welding and the world of robotics. We present two essential aspects of this integration within a CAD modeller: the feature extraction from the CAD model, then an artificial intelligence system which can choose welding parameters according to the experience and the know-how of the welding trade.

First of all we show how it is possible to use information implicitly contained in CAD models of parts to be welded, information at present poorly exploited by CAD-robotic systems. By using the notion of features, we propose a definition of welding paths integrating both geometrical aspects, necessary for the movements of the torch by the robot, and knowledge linked to the welding trade that are indispensable for choosing parameters. A set of methods is proposed in order to automate the paths creation process and to extract the assembly characteristics. The parametric surfaces exploitation allows us to treat a great diversity of welding problems, from the simplest (two flat plates) to the most complex (several parts of complex forms). The process is automated to limit the user's interventions to the choice of parts and joints to weld.

As for the automatic generation of welding parameters (trade aspect) in the framework of robotized GMAW welding, we propose an approach based on the utilization of backpropagation neural networks. It presents many advantages compared with artificial intelligence systems generally used in the welding area (data analysis, expert systems, modelling) by integrating database and modelling aspects and by learning from experience (reference database) with no need for knowledge formalization. The proposed system allows us to take into account quantitative as well as qualitative data. It is thus possible, taking into account the features extracted from the CAD models (positions, orientations, thicknesses and materials composing the assembly), to treat the entire welding problem from the process choice and the electrode choice to the parts preparations (chamfer, plate gap) and determination of the operating parameters. Moreover, the learning method allows us to update the system easily in order to enlarge its initial area according to new knowledge. For example, we can consider other welding processes than these proposed or add new parameters not considered by the current system.

Our work is part of the new concepts developed in the area of CIM, integrated design and CAD. Moreover, it shows the interest of new approaches for some unsolved problems in this area, where experience and know-how are often determining factors for the choice of many design and manufacturing parameters.

References

Alberry, P. J., 1989, Computer model for multipass repair welds in SA508 class 2 alloy. *Welding Journal Research Supplement*, October, 410–417.

Andersen, K., Cook, G. E., Karsai, G., and Ramaswamy, G., 1990, Artificial neural networks applied to arc welding process modeling and control. *IEEE Transactions on Industry Applications*, **26** (5), 824–830.

Andersen, K., Cook, G. E., Springfield, J. F., and Barnett, R. J., 1991, Applications of artificial neural networks for arc welding. *Proceedings of the Conference on Intelligent Engineering Systems Through Artificial Neural Networks*, pp. 717–728.

Barborak, D. M., Dickinson, D. W., and Madigan, R. B., 1991, PC-based expert systems and their applications to welding. *Welding Journal*, January, 29–38.

Bernasek, M., 1991, Software solutions improve productivity. *Welding Journal*, January, 57–60.

Bourret, P., Reggia, J., and Samulides, M., 1991, *Réseaux Neuronaux—Une Approche Connexioniste de l'Intelligence Artificielle* (Paris: Teknea).

Breat, J. L., and Pauwels, J. P., 1991, MIG-expert système expert de génération des paramètres de soudage. *Proceedings of the A2RP* (Actions de Recherche en Robotique and Productique), pp. 223–241.

Breat, J. L., Guezon, M., Pauwels, J. P., and Cotton, C., 1992, MIG-expert système expert de génération des paramètres de soudage. *Soudage et Techniques Connexes*, May-June, 17–21.

Cornu, J., 1985, *Procédés 1: Traité de Soudage Automatique* (Paris: Hermes).

Davalo, E., and Naim, P., 1990, *Des Réseaux de Neurones* (Paris: Eyrolles).

Demuth, H., and Beale, M., 1991, *MATLAB: Neural Network Toolbox, For Use with MATLAB, User's Guide* (New York: The Math Works Inc.).

Detriche, J. M., 1987, *Robots de Soudage, Commande et Programmation, Traité de Soudage Automatique* (Paris: Hermes).

Diday, Lemaire, Pouget, and Testu, 1982, *Éléments d'Analyse de Données* (Paris: Dunod).

Galopin, M., 1989, Une nouvelle approche à l'optimisation du mode opératoire des procédés de soudage. *Soudage et Techniques Connexes*, November-December, 69–73.

Harvas Martinez, C., Romero Soto, E. J., Garcia Pedrajas, N., and Medina Garnicer, R., 1992, Comparison between artificial neural networks and classical statistical methods in pattern recognition. *Proceedings of IPMU 92—Advanced Methods in Artificial Intelligence*, pp. 351–360.

Hascoët, J. Y., Lefoff, O., and Bennis, F., 1994, CAD and artificial intelligence: an application of neural networks to off-line programming for welding robots. *Revue Internationale de CFAO et d'Infographie*, **9** (5), 657–664.

Legoff, O., Hascoët, J. Y., and Bennis, F., 1994a, Application of neural networks to welding parameters generation. *Proceedings of ICARCV'94, 3rd International Conference on Automation, Robotics and Computer Vision*, **2**, 1329–1333.

Legoff, O., Hascoët, J. Y., and Bennis, F., 1994b, Automatic generation of welding parameters with neural networks. *Proceedings of the 12th IASTED International Conference on Application Informatics*, pp. 64–67.

Legoff, O., 1995, Extraction d'attributs et réseaux neuronaux en programmation hors-ligne de robots de soudage. PhD thesis, Université de Nantes et École Centrale de Nantes, France.

Lippman, R. P., 1967, An introduction to computing with neural nets. *IEEE ASSP Magazine*, April, 4–22.

McCulloch, W. S., and Pitts, W. H., 1943, A logical calculus ideas immanent in nervous activity. *Bulletin of Mathematical Biophysics*, **5**, 115–133.

McGlone, J. C., 1979, Submerged arc butt welding of prepared joints. *The Welding Institute Research Bulletin*, February, 41–45.

Nguyen, D., and Widrow, B., 1990, Improving the learning speed of a 2-layer neural network by choosing initial values of the adaptive weights. *Proceedings of the International Joint Conference on Neural Networks*, July 1990, **3**, pp. 21–26.

Riedmiller, M., and Braun, H., 1993, A direct adaptive method for faster backpropagation learning: the RPROP algorithm. *Proceedings of the 1993 IEEE Interntional Conference on Neural Networks*, pp. 586–591.

Rumelhart, D. E., Hinton, G. E., and Williaws, R. J., 1986, Learning internal representations by error propagation. In *Parallel Data Processing*, D. Rumelhart and J. McClelland (eds) (Cambridge, MA: MIT Press), **1** (8), pp. 318–362.

Scalero, R. S., and Tepedelenlioglu, N., 1992, A fast new algorithm for training feedforward neural networks. *IEEE Transactions on Signal Processing*, **40** (1), 202–210.

Sutter, V., and Xu, X., 1993, Welding parameters development using computer modeling. *Proceedings of the 74th American Welding Society Convention*, p. 99.

Vogl, T. P., Mangis, J. K., Rigler, A. K., Zink, W. T., and Alkon, D. L., 1988, Accelerating the convergence of the backpropagation method. *Biological Cybernetics*, **59**, 257–263.