Dimensional Synthesis of Planar Mechanisms Using Neural Networks: Application to Path Generator Linkages

Adrian Vasiliu, Bernard Yannou

Dimensional synthesis of planar mechanisms using neural networks: application to path generator linkages

Adrian Vasiliu, Bernard Yannou *

Ecole Centrale Paris, Laboratoire Productique Logistique, 92295 Châtenay-Malabry, France

Received 15 May 1997; accepted 27 April 2000

Abstract

We propose an original method to synthesize the dimensions of a planar mechanism (linkage) whose function is to generate a trajectory shape. Most graphical and analytical synthesis methods for path generators require specifying the desired trajectory in a non-functional manner, by a list of points rather than a pure shape. Concerning the dimensional optimization methods, they turn out to be slow and their convergence depends on the initial solution. Alternatively, we propose a case-based approach (i.e., couples of trajectories and dimensions of a given structure mechanism) using a neural network. The first stage consists in the generation of a huge case number through kinematic simulations, for random values of dimensions, and in a learning process of the neural network. In the second stage, of utilization, the neural network instantaneously makes it possible to obtain an approximate solution of the synthesis problem, which is an interpolation of close cases. We show on the four-bar linkage example the good quality of the synthesized solutions, for a tiny size of the network. Next, these solutions may be used as judicious initial solutions for a conventional dimensional optimization.

1. Introduction

The path generator synthesis method presented in this paper is a part of a Ph.D. work (cf. Vasiliu [11]), implemented in an integrated predesign platform called REALISME. *Path generators* (see examples in Fig. 1) are subjected to a main kinematic specification: the path described by an *effector point*. This path is considered with or without time dependence, i.e., considering or not the speed along the path. We are concerned here only with time independent synthesis.

Generally, existing methods require the designer to specify the desired path in the form of a list of *precision point* corrdinates. In the case of *analytical synthesis methods* (cf. [9]), the number of

* Corresponding author. Tel.: +33-1-41-131605; fax: +33-1-41-131272.
E-mail addresses: adrian@pl.ecp.fr (A. Vasiliu), yannou@pl.ecp.fr (B. Yannou).

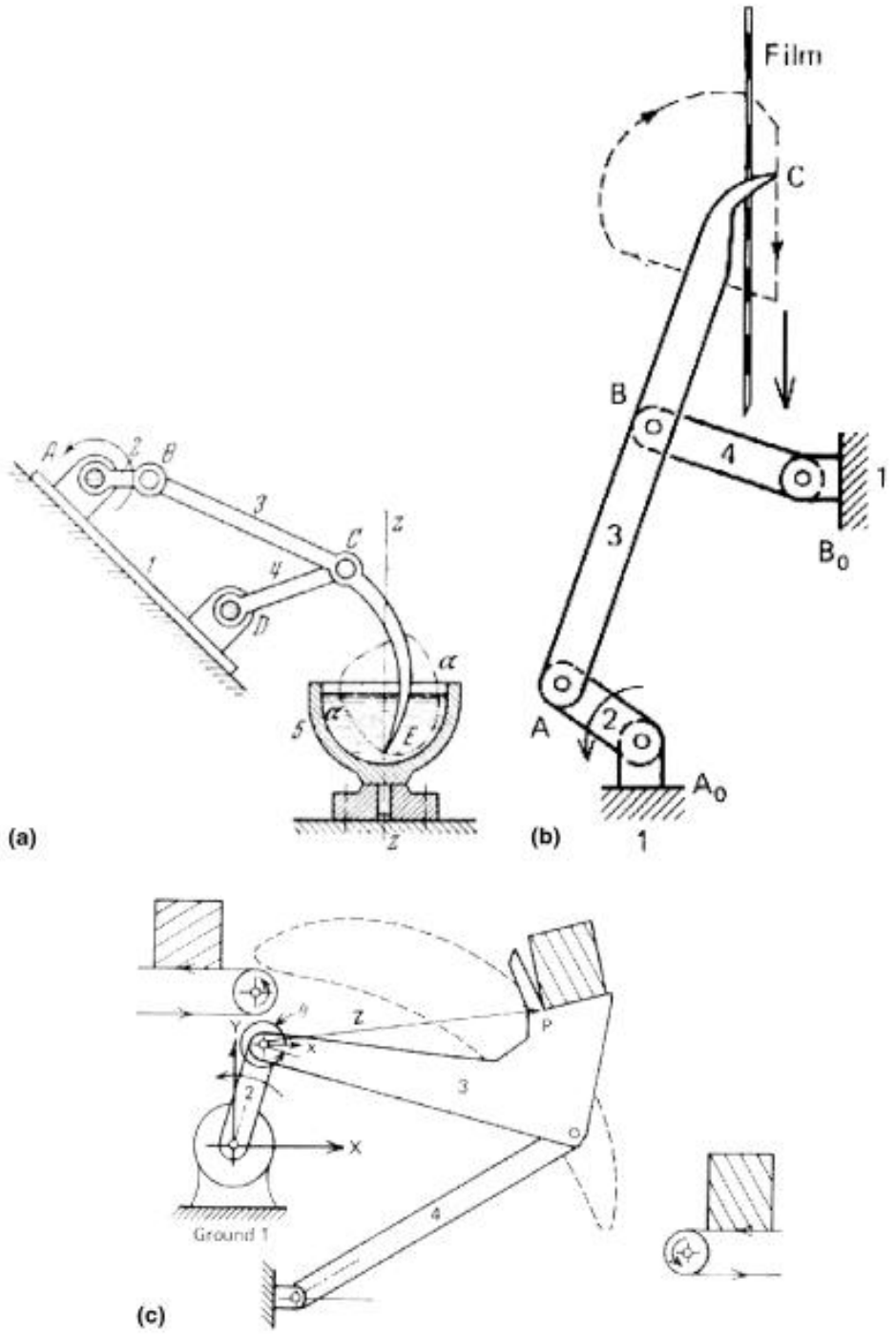


Fig. 1. Industrial examples of path generators (four-bar linkage). (a) Kneading machine. (b) Film indexing mechanism. (c) Mechanism for assembling machine (cf. [9]).

specified points is small (typically between 3 and 10) and depends on the structure. This number may reach a few dozens in the case of *dimensional optimization methods* (cf. [5]). This way of specifying the desired path – imposed by the nature of synthesis methods – is compatible with the designer needs when no particular requirement on the path between these *precision points* is

needed. But, if the real desired function is the *shape* of the path, these methods are not well suited, because they take into account an incorrect synthesis objective and deteriorate the functional specification into a list of points. For this reason, we represent the shape of the path by the Fourier coefficients of his harmonic analysis, by a method proposed by McGarva [7,8]. In this manner, the shape becomes the only objective of the preliminary design.

On the other hand, the dimensional optimization is slow and its convergence is not warranted at all. Probably, these are the reasons why Erdman [3], in his state of the art on the computer-aided mechanism design, considered that an ideal for the future are the tools integrating also a sort of "memory" or "previous solutions library". Some approaches of this kind already exist (cf. [1,6,10,13]), but they use most of the time a simple path atlas (on paper or on computer). There are three major drawbacks: these atlases are incomplete, they are cumbersome and the synthesis result cannot be better than the best mechanism which was recorded in the atlas. In order to eliminate these drawbacks, we propose a synthesis method using a *neural network*. Neural networks roughly represent an analogy of a graph of neurons. There are *input neurons* which code, in our case of path generator, a desired path, and *output neurons* which code the dimensional parameter values of a given planar mechanism. Neurons are connected together with *synaptic connections*, each of them supporting a *synaptic coefficient*. A neuron output value is computed as a function of a weighted sum of input values where the weights correspond to synaptic coefficients. Handling neural networks is generally decomposed into three phases: a generation of design cases (in this work, a design case corresponds to the couple: dimensional parameter values, corresponding simulated path), a learning phase in order to find the best synaptic coefficients, and a utilization phase. During the utilization phase, the neural network is used in the inverse sense, for synthesizing the dimensional parameter values corresponding at best to a desired path. This neural network approach combines three advantages to be related to the three major drawbacks previously expressed: speed of the synthesis phase, small size of the resulting neural network and interpolation of close cases. The limits of this approach are also discussed.

2. Synthesis using neural networks

2.1. Case-based reasoning

Neural methods can be considered as a particular implementation of *case-based reasoning*. A *case* of mechanism (in this context) has a *content* which can be reached using an *index*. The content represents the structure of the mechanism (dimensions and morphology) generating a path which is the index. For a given morphology and a given dimension set, we are able to determine the trajectory, by simulation, and in the form of a list of points. Consequently, there is a mathematical function (see Fig. 2(a)) from the mechanism (morphology and dimensions) space to the trajectory space.

Formulated in this way, the joint morphological and dimensional synthesis problem may be seen as the inversion of this function (see Fig. 2(b)) giving the morphology and the dimensions for a given desired path. Unfortunately, the inverse function does not exist, generally, because the same path shape may be described by several mechanisms of different morphologies and

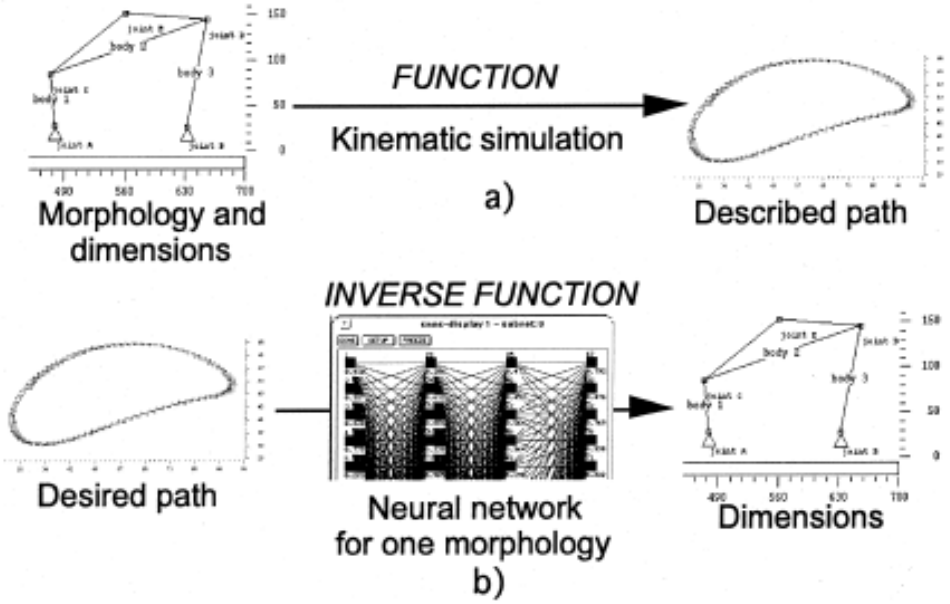


Fig. 2. (a) The first stage: case generation using kinematic simulation, and learning of the neural network. (b) Second stage: neural network utilization.

dimensions. This is an important limitation of the function approximation for the path generator synthesis, but this problem will be partially solved by a judicious choice of the dimensional parameters and of their value intervals.

2.2. *Artificial neural networks*

The mathematical method of function approximation used in REALISME is an *artificial neural network*. [1] The latter is a graph of *neurons* including input and output neurons. In our case, input neurons represent the shape of the path, and the output neurons represent the dimensional parameter values of the mechanism (*content*).

We found no sense in interpolating morphologies because there is no continuity between different morphologies. Moreover, the representation of the morphology of a mechanism (usually as a graph) requires a large amount of data and we did not know a way to represent a morphology with a given number of real values as required by a neural networks. For all these reasons, the method used in REALISME consists in a loop over several neural networks' requests, each neural network corresponding to a particular mechanism morphology. So, only the dimensions are represented in the network, not the morphology. Practically, these networks are created for the simplest or the cheapest morphologies.

When interpolating dimensions, we hope that the synthesized mechanism will have a better described path (comparatively to the desired path) than any case previously learned by the neural

[1] The stuttgart neural network simulator (SNNS) is used.

network. This is generally the case but, because of the non-inversion of the mathematical function (mechanism → path) (see Fig. 2(b)), an interpolation can lead to a worth solution than any of close cases.

2.3. Case representation

The index and the content of a case must be as *little as possible* in order to reduce the dimension of the problem. This is the reason for the use of *normalization* procedures (Section 3.3), which ensure that each case represents an entire class of mechanisms, whose path forms are equivalent. Other constraints are imposed by the choice of a neural network to store and interpolate cases: data of numerical type only (not symbolic), with continuous values (not discrete), scaling of data, constant cardinality coding. This last constraint will be entirely respected by our choice of path coding with a fixed number of Fourier coefficients.

3. Coding a path

3.1. Indexation procedure

By *indexation of cases*, we understand a transformation (or coding), before storage, of the data which represent the path of a case, in order to reduce its size and to allow its utilization. The procedure used in REALISME consists in a series of transformations, which will be described here after.

3.2. Harmonic analysis of paths

We use (with some improvements) the model of McGarva [7,8], which makes it possible to approximate the shape of a path by the complex coefficients of the first five harmonics, resulting from the decomposition of a path in a Fourier series. This is done from the representation of the path by a set of points coordinates, obtained by kinematic simulation. The position of a point on the path may be represented by a complex function: $z(t) = x(t) + \mathrm{i}y(t)$. For closed paths, the function is periodic. The Fourier development is then $z(t) = \sum_{m=-\infty}^{\infty} a_m \exp(2\pi \mathrm{i} m t)$. The complex coeffcients of Fourier are $a_m = \int_0^1 \exp(-2\pi \mathrm{i} m t) z(t)\, \mathrm{d}t$. The a_0 coefficient is called fundamental, a_1 and a_{-1} represent the first harmonic, a_2 and a_{-2} the second harmonic, etc. Because the function $z(t)$ is known only at the particular points z_k $(k = 1, \ldots, N)$, the integrals are discretized using the trapeze formulas (t_k is the curvilinear abscissa of the point k): [2]

$$a_m = \frac{1}{N} \sum_{k=0}^{N-1} \frac{t_{k+1} - t_k}{2} \left(z_{k+1} \exp(-2\pi \mathrm{i} m t_{k+1}) + z_k \exp(-2\pi \mathrm{i} m t_k) \right). \tag{1}$$

[2] With this parameterization of the path, the information linked to the speed is eliminated, which is important for a time independent synthesis method.

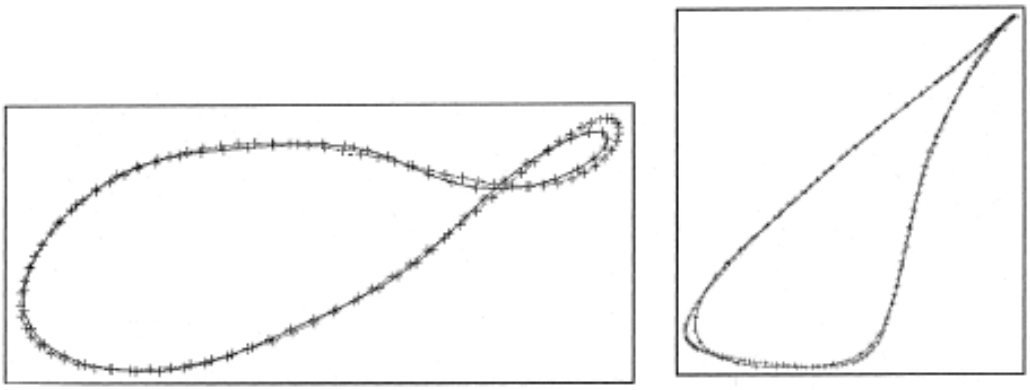


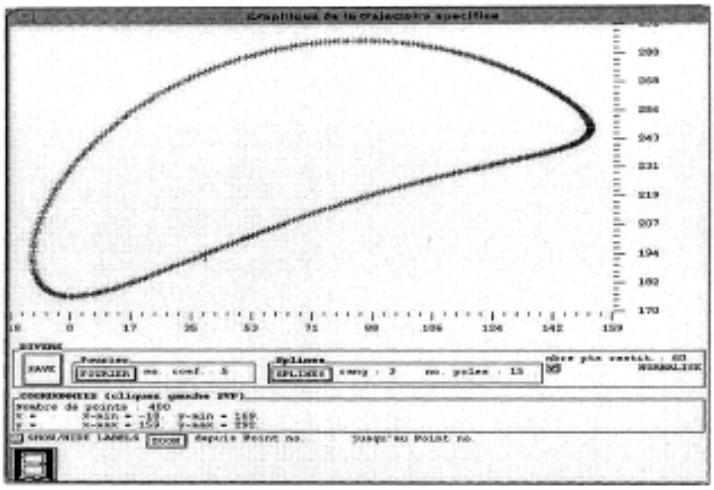
Fig. 3. Examples of the approximation of a path by the first five harmonics. The desired path and its corresponding synthesized path (by the first five harmonics) are superimposed.

Fig. 3 presents two examples of path approximation. The differences between the original path and its approximation by the first harmonics are significant only in the zones with a big curvature or angular points. The choice of a number of five harmonic seems to us a good compromise.

3.3. Normalization

The invariance of the link between path shape and mechanisms in relation to linear transformations allows – thanks to a specific *normalization procedure* – to obtain a sort of shape equivalence class. The Fourier coefficients of the normalized path are representative to this class. The objective of the normalization is to obtain a representation of the shape, independent of linear transformations which may occur: translation, rotation, scale, direction of point listing. Moreover, we must make abstraction of the arbitrary first point of the circular list point.

Our normalization procedure is an improved version of those proposed by McGarva [7,8]. Globally, the procedure is based on the fundamental harmonic and the first harmonic. For



	Before normalization		After normalization	
m	$\mathrm{Re}(a_m)$	$\mathrm{Im}(a_m)$	$\mathrm{Re}(a_m)$	$\mathrm{Im}(a_m)$
-5	0,65	1,26	-0,01	0,00
-4	0,94	-0,34	0,00	0,02
-3	5,26	0,32	0,00	0,00
-2	2,88	2,59	0,01	0,10
-1	-11,51	-57,69	0,34	0,00
0	63,44	235,91	0,00	0,00
1	-17,74	9,91	0,00	0,00
2	-6,13	-0,26	0,09	0,07
3	-0,41	-0,17	0,09	0,00
4	0,44	1,14	0,00	-0,02
5	-0,20	0,24	0,02	0,00

Fig. 4. Examples of paths and Fourier coefficients for the first five harmonics.

example, the coefficient a_0 represents the position of the center of gravity of the path. Translating this center at the origin of the Cartesian coordinate system $(0,0)$, by the transformation $a_0 \leftarrow 0$, we obtain a translation normalized position. Similar procedures are used for the other transformations. The normalization eliminates five Fourier coefficients and reduces the total number from 22 to 17 normalized coefficients. An example is given in Fig. 4.

4. Coding the dimensions of the mechanism

The mechanisms dimensions may be described by different parameterizations. REALISME uses absolute Cartesian parameterizations using the link coordinates in the initial position of the mechanism, which ensures unicity of definition (only one possible assembling) and the homogeneity of variables (no mix of coordinates and angles when only rotational joint are used). Moreover, the mechanism can thus always be assembled in the initial position and the described path contains therefore at least one point.

For the examples of the *four-bar mechanism*, is described (without further explanations) (see Fig. 5) a reasonable choice of the value intervals for the five-dimensional parameters. Indeed, in order to obtain, using a neural network, an approximation of the function going from the dimensions to the paths, the choice of the dimensional parameters and of the intervals for each parameter is subject to the constraints already mentioned: limitation of the zones where the function is not invertible and of the zones where the mechanism blocks during the simulation. [3] The last constraint would not exist if it was possible to approximate both closed and open paths. Unfortunately, we do not know such a representation which passes continuously from closed paths to open paths. Moreover, the harmonic analysis works for periodic functions, i.e. closed paths, whereas splines or nurbs approximation is specific to open paths, because it is necessary to randomly chose a cutting point on the path. In definitive, we have chosen to tackle the synthesis of closed paths.

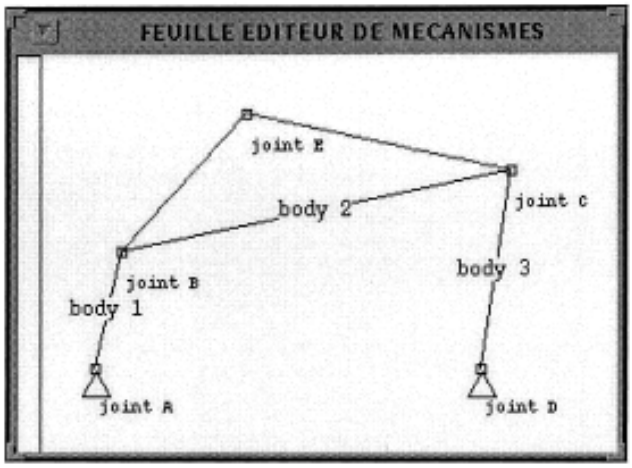
5. Learning of the neural network

The complexity of the function whose approximation we search for imposes the utilization of a big amount of value samples. The random choice of dimensions imposes the use of a big number of value samples. The random choice of dimensions is preferable – because of the non-linearity of the function – to a regular scanning (with constant interval).

5.1. Dimensions of the neural network

The sturcture of the neural network used is a classical, feedforward one, with 17 input neurons corresponding to the normalized Fourier coefficients; 2 hidden layers of 22 neurons each; 5 output

[3] We do not have at our disposal a general procedure allowing to determine the feasible intervals for each mechanism morphology, even if we have the non-blocking constraints for some morphologies, as the Grashov conditions for the *four-bar* mechanism.



$x_A = 0$	$y_A = 0$
$-40 \leqslant x_B \leqslant -5$	$y_B = 0$
$0 \leqslant x_C \leqslant 55$	$5 \leqslant y_C \leqslant 100$
$x_D = 100$	$y_D = 0$
$0 \leqslant x_E \leqslant 100$	$0 \leqslant y_E \leqslant 100$

Fig. 5. The choice of the parameters for the four-bar mechanism. The motor is in A and the effector point is point E.

neurons, corresponding to the dimensional parameters of the four-bar mechanism. The activation function of the neurons (with the exception of the input layer) is of a classical shape: $f_{\text{act}}(x) = 1/(1+\mathrm{e}^{-x})$.

In total, 60,000 cases were generated by kinematic simulations for the four-bar mechanism. In a conventional manner, the cases were divided into three categories: 30,000 as *learning patterns* (used to calculate the error function during the learning process), 15,000 as *validation patterns* (used to estimate, during learning, the evolution of the generalization ability of the network) and 15,000 as *test patterns* (used to test the generalization capacity on data never used during learning).

5.2. Scaling of inputs and outputs

The mathematical nature of the activation function used in the network requires that input and output values are from 0 to 1. A specific procedure of scaling is employed, which takes into account the minimum and maximum values of the neurons. For the input neurons, the extreme values are determined for all the cases, independently for each input neuron. For the output neurons, this determination is done globally, for all the output neurons. So, if $e_i^{(k)}$ and $s_i^{(k)}$ are the ith input, respectively, output for the kth case, the scaled corresponding values $e_i^{(k)\prime}$ and $s_i^{(k)\prime}$ are calculated thus (n_e = number of inputs, n_s = number of outputs and N = total number of cases):

$$
\begin{aligned}
e_i^{(k)\prime} &= \frac{e_i^{(k)} - \min\limits_k\left(e_i^{(k)}\right)}{\max\limits_k\left(e_i^{(k)}\right) - \min\limits_k\left(e_i^{(k)}\right)}, \quad i = 1,\ldots,n_e \text{ and } k = 1,\ldots,N, \\
s_j^{(k)\prime} &= \frac{s_j^{(k)} - \min\limits_j\left(\min\limits_k\left(s_j^{(k)}\right)\right)}{\max\limits_j\left(\max\limits_k\left(s_j^{(k)}\right)\right) - \min\limits_j\left(\min\limits_k\left(s_i^{(k)}\right)\right)}, \quad j = 1,\ldots,n_s \text{ and } k = 1,\ldots,N.
\end{aligned}
\tag{2}
$$

5.3. Learning algorithm

Many learning algorithms for the neutral network were tested, including global methods such as *MonteCarlo* and *simulated annealing*; The Resilient backpropagation (Rprop) algorithm is the one which gives the best results. Using the learning patterns, learning was done in 14,000 iterations. Total learning time was about 30 h, [4] with a quadratic approximation error E of less than 0.02. The approximation error is given by the formula (t_{js} is the output value of the network and o_{js} is the output value contained in the patterns): $E = \sum_{j \in \text{pattern}} \sum_{s \in \text{output\_var}} (t_{js} - o_{js})^2$.

6. Neural network utilization

6.1. Procedure

In the utilization phase, the designer has at his disposal some networks, corresponding to different morphologies of mechanisms. The steps, going from the specification of the desired path shape to the synthesis of the mechanism, will be detailed here after. The specification of the desired path shape is done in REALISME with a special path editor. [5] With the mouse, control points of the spline curve are defined. Equal distributed points on this curve are determined for the use in the harmonic analysis. The curve approximated by its first five harmonics is presented to the designer, who decides if the approximation is acceptable or not.

Next, the indexation procedure is the same as for the construction of the case base (see Section 3), and the result is a series of scaled normalized Fourier coefficients. Minimal and maximum values used for scaling are those stored, for the same coefficient, during the creation of the base (see Section 5.2). This procedure is done successively for each of the available networks, because the extreme values are a priori different.

Finally, the network gives an interpolated solution, i.e., the output values (transformed again in order to eliminate the effect of scaling) which represent the dimensions of the synthesized mechanism. Using these values to determine the initial position of the mechanism, the kinematic simulation is automatically done. The path described by the synthesized mechanism and the desired path are drawn in the same window (see Fig. 6) in order to let the designer decide on the quality of the resulting mechanism. In most of the cases, we were visually satisfied by the similarity of the shape of those paths. In spite of that, the problem of judging the relevance of the method is not simple. Indeed, we have to dissociate between the quality of the method (neural approximation) and two types of error. The first error is due to the impossibility to calculate the inverse function, already mentioned, for which the interpolation is not correct. The second error is simply

[4] On a PC 90 MHz computer.

[5] Another possiblity would be the utilization of the approach of *declarative modelling of curves* (cf. Daniel and Lucas [2]) in order to put in the hands of the designer an interface to specify forms in a natural (qualitative) language. This declarative specification would then be translated into some "probable splines" which could feed our own procedure to calculate the Fourier coefficients.

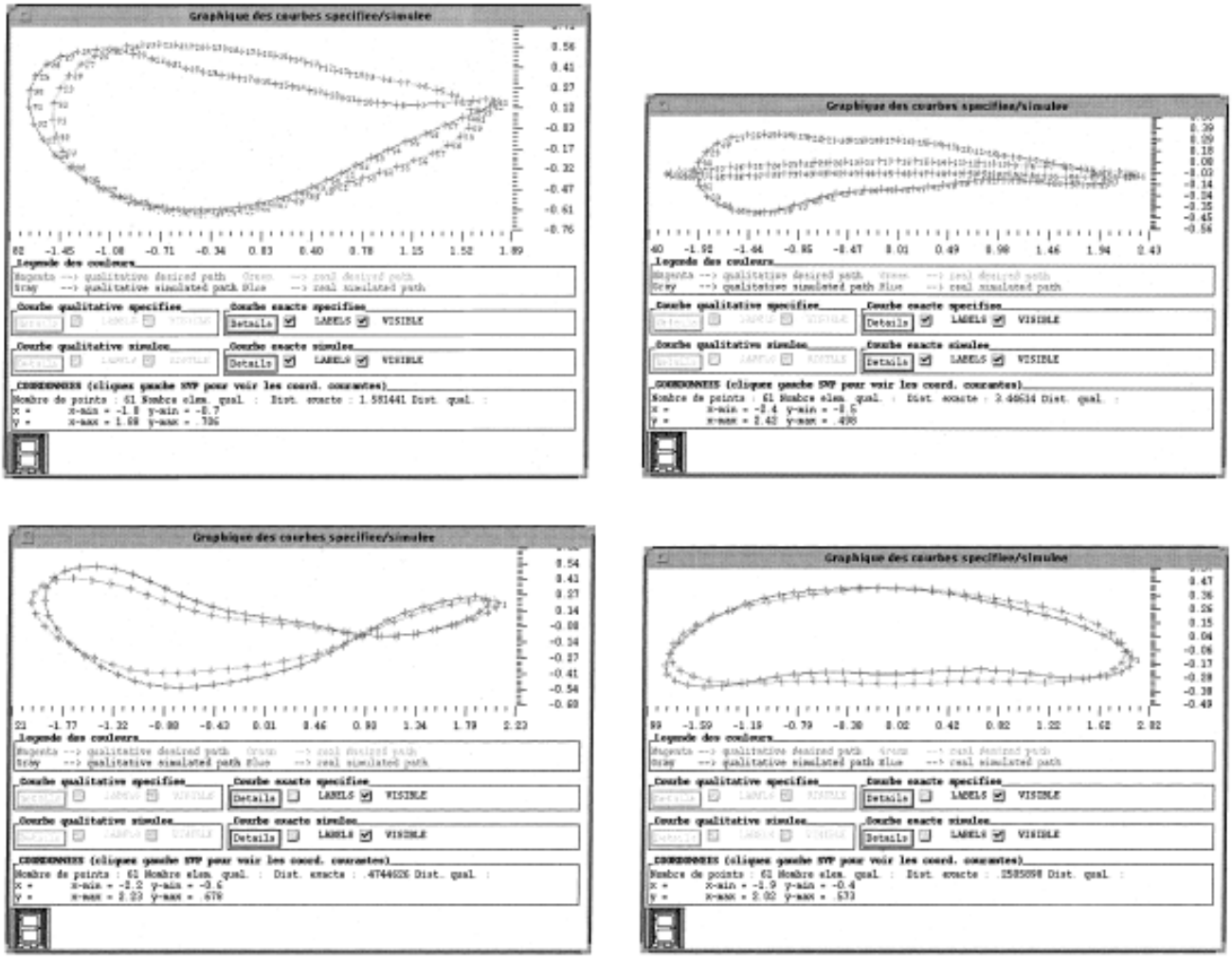


Fig. 6. Examples of synthesized paths for four desired paths randomly chosen. The desired path and the synthesized path are drawn in the same window.

due to the fact that a mechanism, with a given morphology, cannot generate all paths (for example, algebraic curves of 6th degree for the four-bar mechanism). Nonetheless, we are now working to create more rigorous test procedures.

6.2. *Adaptation of the synthesized solution*

The last step consists of an improvement of the synthesized case, classically called *adaptation* (to the specification). Two methods are proposed. The first method is based on the only link characterization between dimensions and paths. In other words, a linear transformation (translation, rotation, scale) applied to the mechanism has the same effect on the path. So, we define a method of *geometrical calibration*, whose principle consists in the determination of the optimal transformation allowing to superimpose at best the desired and synthesized paths. An original procedure, based on the normalization of the paths, was developed. A second method of adap-

tation consists in employing the synthesized solution (when it is not entirely satisfactory) as an initial solution of a *dimensional optimization*.

7. Conclusion

The main idea of our path generator synthesis method is the memorization of the link between geometry and the path shape using a "compilation" of the mechanisms by neural networks. This method has two main advantages. First, this method take into account shape specifications, which is very difficult with traditional methods. So, in the case of a path catalogue (atlas), this is possible only by a visual choice (as in Zhang et al. [13]) or using very expensive algorithms in order to compare the form of the paths (as in Kota and Chiou [4]). Second, the preliminary "compilation" of the base of cases using the interpolation with a neural network enormously reduces the size of the base. In this manner, it becomes possible to provide on a single floppy disk a set of neural networks corresponding to different morphologies. For example, the size of the neural network for the four-bar mechanism is less than 1 kB, while the size of the record of all the paths used in the learning process is 12 MB (which corresponds to the size of a classical catalogue/atlas).

The implementation of this method requires a big number of kinematic simulations. So, it is particularly useful to have at one's disposal a very rapid simulation method. This condition is achieved in REALISME by the use of the *systemic multipolar approach* (which have a *parametric nature* and minimize the iterative calculations using close-form solutions most of the time; for details, see [11,12]).

Various evolutions are possible: the use of advanced neural network techniques, the extension to time dependent paths, the adaptation to open paths using spline function in place of Fourier series, finally the creation of a neural networks database for a representative set of mechanisms.

References

[1] C. Bagci, D. Burke, Optimum synthesis of coupler curve and uniform rotary motion driven multiloop mechanisms generating complex output motions, J. Mech. Design 115 (12) (1993) 967–977.

[2] M. Daniel, M. Lucas, Towards declarative geometric modeling in mechanics, in: Proceedings of the IDM-ME'96, Ecole Centrale de Nantes, 15–17 Avril 1996, pp. 455– 464.

[3] A.G. Erdman, Computer-aided mechanism design: now and the future, Trans. ASME, J. Mech. Design, Special 50th Anniversary Design Issue 117 (1995) 93–100.

[4] S. Kota, S.-J. Chiou, Use of orthogonal arrays in mechanism synthesis, Mech. Mach. Theory 28 (6) (1993) 777–794.

[5] S. Krishnamurty, D.A. Turcic, Optimal synthesis of mechanism using nonlinear goal programming techniques, Mech. Mach. Theory 27 (5) (1992) 599–612.

[6] M. Mayourian, F. Freudenstein, The development of an atlas of the kinematic structures of mechanisms, J. Mech. Trans. Automation Design 106 (12) (1984) 458–461.

[7] J. McGarva, G. Mullineux, Harmonic representation of closed curves, Appl. Math. Model. 17 (4) (1993) 213–218.

[8] J.R. McGarva, Rapid search and selection of path generating mechanisms from a library, Mech. Mach. Theory 29 (2) (1994) 223–235.

[9] G. Sandor, A. Erdman, Advanced Mechanism Design: Analysis and Synthesis, vol. 2, Prentice-Hall, Englewood Cliffs, NJ, 1984.

[10] V. Unruh, P. Krishnaswami, A computer-aided design technique for semi-automated infinite point coupler curve synthesis of four-bar linkages, J. Mech. Design 117 (1995) 143–149.

[11] A. Vasiliu, Une approche CAO pour la préconception des mécanismes plans générateurs de trajectoire: REALISME. Thèse de Doctorat, Ecole Centrale Paris, 1997.

[12] B. Yannou, A. Vasiliu, Design platform for planar mechanisms based on a qualitative kinematics. In QR'95: Ninth International Workshop on Qualitative Reasoning about Physical Systems, Amsterdam, Pays-Bas, 18–19 Mai 1995.

[13] C. Zhang, R.L. Norton, T. Hammond, Optimization of parameters for specified path generation using an atlas of coupler curves of geared five bar linkages, Mech. Mach. Theory 19 (1984) 459–466.