Optimization of the global matching between two contours defined by ordered points using an algorithm based on dynamic programming

FRANCISCO P. M. OLIVEIRA[1]
JOÃO MANUEL R. S. TAVARES[2]

FEUP - Faculdade de Engenharia da Universidade do Porto
INEGI - Instituto de Engenharia Mecânica e Gestão Industrial
Rua Dr. Roberto Frias, 4200-465 Porto, PORTUGAL
[1]franciscopmoliveira@yahoo.com.br
[2]tavares@fe.up.pt

Abstract. This paper presents a new assignment algorithm with order restriction, developed using the paradigm of dynamic programming. The algorithm was implemented and tested to determine the best global matching between two sets of points that represent the contours to be matched. In the experimental tests done, we used the affinity matrix obtained via the method proposed by Shapiro based on geometric modeling and modal matching.

The proposed algorithm revealed an optimum performance, when compared with the classic assignment algorithms considered in this work: Hungarian method, Simplex for Flow Problems and LAPm. Indeed, the quality of the matching improved when compared with these three algorithms, because the crossed matching, allowed by the conventional assignment algorithms, disappeared. Besides, the computational cost of our algorithm is very low in comparison with the other three, resulting in lesser execution times.

Keywords: Image analysis, contours matching, optimization, dynamic programming.

(Received December 03, 2007 / Accepted March 15, 2008)

1 Introduction

The recognition of objects represented in images is one of the central problems in Computational Vision. It is a challenging task, mainly due to the large number of variations of projection of objects in 2D images; for instance, due to changes of the positions of the cameras used, or even because of deformations that the objects might suffer.

Highly related with the problem of recognition of objects represented in images is the problem of identifying homologous elements in shapes, which are usually defined by groups of points.

The problem of determining the correspondences among characteristic points of an object in two different instants, or between two similar objects, represented in images, originated the emergence of many proposals, in the sense of accomplishment the best global correspondence among the referred points, [18], [16], [20], [9], [14], [4], [3], [13].

Common approaches for shape matching in images consist in dividing the matching process in three steps. In the first step, a set of feature points from each shape is extracted, for example, using an edge detector on each image and then sampling some points from the edges found. In the second step, pairs of corresponding feature points are determined from the two feature points sets. Finally, in the last and third step, the correspondence information obtained is used to find an alignment transformation.

Finding the correspondence between points has been a subject of large and hard research. A usual approach is to build a cost matrix that represents the similarity between all possible pairs of points on the two shapes. In this case, the matching problem can be interpreted as an optimization problem, where the objective function to minimize is the sum of all costs associated to the defined matches. In these approaches, the larger is the value of the sum, the larger is the difference between the shapes considered. Usually, assignment algorithms are used to determine the best global matching. Such algorithms are frequently based on linear or integer programming, [2], [3]; bipartite graph matching, [2], [3], [5], [15]; dynamic programming, [17]; convex optimization, [11]; simulated annealing, [19]; etc.

In the optimization of the correspondences between two closed contours, each one defined by a set of ordered points, could be included an important restriction: *the relative order of the points to be matched should be maintained to guarantee the coherence of the matching obtained*. This restriction guarantee that crossed matches are not allowed.

Initially, this problem of determining the global matching of minimum cost that respects the order of the points became difficult to solve, because this order is not absolute; that is, there are different ordinations that define the same contour. In this work we present the solution developed that is based on dynamic programming and solves this problem in a simple and fast way.

To trial and to compare the new dynamic programming algorithm developed with other usual assignment algorithms, the first one was integrated in a computational platform, already existent, [20], [8]. The results of the comparison with the Hungarian method, Simplex for Flow Problems and LAPm algorithms are also presented in this paper. The cost matrix used for the comparison was obtained using the modal matching methodology proposed by Shapiro, [18], also already implemented in the referred platform, [20], [8]. However, another kind of matching cost matrix could be used.

In this paper, after a reference to some previous works developed to determine the best global matching between two shapes, we approach the problem to determine the best correspondence between two sets of ordered points that respect the order defined.. Afterwards, comparative results between the developed algorithm and the classic assignment algorithms already referred are presented. The last section is dedicated to some final conclusions and future work perspectives.

2 Previous work

This work comes in the sequence of the project presented in [20], where methodologies for matching characteristic points of two shapes in images were implemented, using physical and geometric modeling, complemented with modal matching, [18], [16]. Thus, those methodologies were used to determine the matching between characteristic points from two shapes, through the construction of an affinity matrix. Later, this cost matrix was used to determine the desired correspondences. The solution presented to search for the matching had a pure local nature, in the sense that two points were only corresponded if, for each one of the two, the other point was also the best candidate in terms of matching cost. However, with this local approach it frequently happened that some points were not successfully corresponded and sometimes crossed matching occurred, see Figure 1.

Figure 1: Matching found between two contours "*heart5*" and "*heart6*", using a local search for the correspondences. The contours are defined by 81 and 83 points, respectively.

Later, it was used the work previously referred, [20], and implemented three global optimizations methods to determine the best matching solution between two shapes defined by points, [2], [3] . In this new approach, the problem of determining the best global matching was formulated as a classic assignment problem, being used three algorithms traditionally employed to solve this kind of problem, [1]: the usual Hungarian method, [6], the Simplex for Flow Problems, [10], and the LAPm, [21]. The results obtained, when those three assignment algorithms were applied to the affinity matrix calculated using physical or geometric modeling, had enhanced quality in comparison with the ones obtained using the previous methodology based in pure local aspects, [2], [3].

As already referred, when the assignment algorithms were applied to match contours defined by ordered point sets, we verified that, occasionally, the matching found appeared without sense; that is, the order of the points was not respected and, this way, crossed matches were obtained, see Figure 2. Thus,

the work here presented had as a main objective to develop an assignment algorithm that respects the predefined orders of the points that define the two contours to be matched.

Figure 2: Matching of the contours of Figure 1 using global optimization.

3 Definition of the problem

Let us begin by defining what means in this work *relative order* and *absolute order* of the points that define a contour. From Figure 3, it can be extracted the sequence of points: 1, 3, 4, 6, 7, 9. This sequence is monotonous increasing. Considering the same figure, it can also be extracted the sequence: 4, 6, 7, 9, 1, 3. However, this last sequence is not monotonous.

Considering the Figure 3 as a closed contour, it can be observed that the two previous sequences define exactly the same contour. The difference between the two is only the initial point considered. In this paper, we will say that the first sequence respects the absolute order, because it is monotonous increasing, and that the second one just respects the relative order.

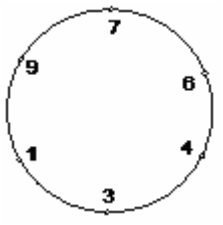

Figure 3: Sequence: 1, 3, 4, 6, 7, 9 placed on a circumference. The same points of the circumference can also be represented, for instance, by the sequence: 4, 6, 7, 9, 1, 3.

To illustrate our solution for the problem of matching the points of two contours maintaining the relative order of the matched points, let us begin to analyze the two following examples:

1. Suppose that we have two contours, both defined by 4 points and numerated from 1 to 4, and consider the next correspondences (given by columns):

$$f = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 1 & 2 & 3 & 4 \end{pmatrix}$$

and

$$g = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 3 & 4 & 1 & 2 \end{pmatrix}.$$

When we observe the second line, which corresponds to the second contour, we can conclude that the correspondence *f* satisfies the absolute order but the correspondence *g* does not. However, the relative order is correct in both, because after point 1 comes point 2, after point 2 comes point 3 and so forth (considering the sequence of points in circle).

2. Suppose now that we have two contours, one defined by 4 points and the other defined by 7 points, respectively. Observe the next correspondences:

$$h = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 1 & 2 & 5 & 7 \end{pmatrix},$$

$$t = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 2 & 4 & 6 & 1 \end{pmatrix}$$

and

$$p = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 6 & 7 & 4 & 5 \end{pmatrix}.$$

All respect the relative order, but only correspondence *h* respects the absolute order.

When equal number of points defines the contours, the matching can be easily accomplished. It is enough to observe that if point *i* of contour 1 corresponds to point j of contour 2, then point $i+1$ of contour 1 has to correspond to point $j+1$ of contour 2, and so forth. This way, considering that each of the two contours is defined by *n* points; there exist just *n* hypotheses of global matching maintaining the relative order:

$$\begin{pmatrix} 1 & 2 & 3 & \ldots & n \\ 1 & 2 & 3 & \ldots & n \end{pmatrix},$$

$$\begin{pmatrix} 1 & 2 & 3 & \ldots & n \\ 2 & 3 & 4 & \ldots & 1 \end{pmatrix},$$

$$\begin{pmatrix} 1 & 2 & 3 & \ldots & n \\ 3 & 4 & 5 & \ldots & 2 \end{pmatrix},$$

$$\ldots,$$

$$\begin{pmatrix} 1 & 2 & 3 & \ldots & n \\ n & 1 & 2 & \ldots & n-1 \end{pmatrix}$$

According to what was briefly explained, it is enough to calculate the cost of each one of the *n* global correspondences and to choose the one that originated a minimum cost.

For contours defined by different or equal number of points, we will present, afterwards, a new formulation based in dynamic programming, which finds the best global matching maintaining the absolute order of the matched points. Then, through the rearranging of the matched points and the comparison of the costs, the best global matching that respects the relative order will be obtained.

4 Formulation as a dynamic programming problem

4.1 General formulation

Let us begin this section with a simple example. Let us suppose that we have contour 1 and contour 2 defined, respectively, by 4 and 6 points and the following cost matrix of the matches between them:

$$C = \begin{bmatrix} 1 & 0 & 1 & 4 & 5 & 1 \\ 0 & 3 & 1 & 5 & 2 & 1 \\ 6 & 1 & 2 & 4 & 0 & 8 \\ 3 & 2 & 7 & 5 & 4 & 1 \end{bmatrix},$$

where c_{ij} represents the cost to match point *i* from contour 1 with point *j* from contour 2.

To avoid the crossed matching, we require that the absolute order of the matched points must be preserved. This way, we impose the monotony of the matching sequence; that is, if point *i* of contour 1 corresponds to point *j* of contour 2, then point $i+1$ (here, $i+1$ means the point that follows point *i* in the sequence of points disposed in circle) of contour 1 must correspond to a point $j+k$ of contour 2, where *k* is integer and $k \geq 1$. Thus, we have, for instance, among others, the following valid correspondences:

$$\begin{pmatrix} 1 & 2 & 3 & 4 \\ 1 & 2 & 3 & 4 \end{pmatrix},$$

$$\begin{pmatrix} 1 & 2 & 3 & 4 \\ 1 & 3 & 4 & 5 \end{pmatrix},$$

$$\begin{pmatrix} 1 & 2 & 3 & 4 \\ 2 & 3 & 4 & 6 \end{pmatrix}$$

and

$$\begin{pmatrix} 1 & 2 & 3 & 4 \\ 3 & 4 & 5 & 6 \end{pmatrix}$$

with respective global costs: 11, 10, 6 and 7.

In total, for the imposed hypotheses, we have exactly 15 possible correspondences, because to count the matching hypotheses is equivalent to count how many subsets of 4 different elements we can get from the 6 elements of contour 2. Therefore, the number of global correspondences that respect the absolute order is in this example given by:

$$C_4^6 = \frac{6!}{(6-4)!\,2!} = 15.$$

In general, if a contour is defined by *n* points and the other by *m* points, with $n \leq m$, there are exactly C_n^m (combinations of *n* elements in a set of *m* elements) matching hypotheses respecting the absolute order. With regard to relative order, there are exactly mC_n^m hypotheses, as we will explain later. In [13] is presented an optimal solution, with order restriction, to the matching contours defined by the same number of points.

Using a usual notation in dynamic programming, [12], [22], for the previous example, we will define 4 stages. In stage 1, the matching of smaller cost for point 1 of contour 1, subject to the matching hypotheses is chosen. In stage 2, the best matching for point 2 of contour 1 is selected, subjected to the matching hypotheses derived from the matching of point 1 in stage 1, and so forth. It is fundamental to refer that the definition of a matching between two points in a certain stage will affect the hypotheses of matching in the following stages.

To help us understand the situation previously described, let us observe the following. In the example in study, point 1 of contour 1 can just match the points 1, 2 or 3 of contour 2; but, for instance, if point 1 of contour 1 matches point 3 of contour 2, then point 2 of contour 1 has only one hypothesis to match – point 4 of contour 2. Thus, and according to the matching already effectuated in the previous stages, for a certain stage *k* from the example in study, point *k* of contour 1 will match just a point of the following groups of points of contour 2: $\{k\}$, $\{k,\ k+1\}$ or $\{k,\ k+1,\ k+2\}$.

To indicate if a point of contour 1 has 1, 2, or 3 points of contour 2 available for matching, we will define the state variable *s*. For the previous example, we have $s \in \{1,2,3\}$. If in a certain stage *k* we have $s=1$, then point *k* of contour 1 has only one matching hypothesis (with point *k* of contour 2); if $s=2$, then point *k* of contour 1 has two matching hypotheses (with points *k* or $k+1$of contour 2), and so forth.

Let us define now the function of minimum cost $f_k(s)$, where *s* is the state variable already defined, *k* represents the stage and $f_k(s)$ represents the minimum cost to correspond points 1, 2, 3, *k* of contour 1, when point *k* of contour 1 has *s* matching hypotheses to choose.

To best clarify our approach, we will apply this formulation to the example in study. Thus, we will build, successively, an optimal correspondence that respects the absolute order of the points. For such, on the left side of the arrow we indicate the minimum costs for each stage and for each state, and on the right side of the arrow we define the correspondence:

$$f_1(1) = c_{11} = 1 \rightarrow \begin{pmatrix} 1 \\ 1 \end{pmatrix}$$

$$f_1(2) = \min\{c_{11},\ c_{12}\} = 0 \rightarrow \begin{pmatrix} 1 \\ 2 \end{pmatrix}$$

$$f_1(3) = \min\{c_{11},\ c_{12},\ c_{13}\} = 0 \rightarrow \begin{pmatrix} 1 \\ 2 \end{pmatrix}$$

$$f_2(1) = c_{22} + f_1(1) = 3 + 1 = 4 \rightarrow \begin{pmatrix} 1 & 2 \\ 1 & 2 \end{pmatrix}$$

$$f_2(2) = \min\{c_{22} + f_1(1),\ c_{23} + f_1(2)\} = 1 \rightarrow \begin{pmatrix} 1 & 2 \\ 2 & 3 \end{pmatrix}$$

...

$$f_4(3) = \min\{...\} = 2 \rightarrow \begin{pmatrix} 1 & 2 & 3 & 4 \\ 2 & 3 & 5 & 6 \end{pmatrix}$$

As in the total there are 4 stages, if it is just intended to calculate the minimum cost, in the fourth stage it would not be necessary to calculate $f_4(1)$ and $f_4(2)$ but, because it is necessary to keep relative information about the correspondence, such has to be done. Thus, we have that the minimum cost to match the 4 points of contour 1 with 4 points of contour 2, respecting the absolute order of the points, is 2 and the associated correspondence is the last suitable.

In general, for a costs matrix *C* of dimension $n \times m$, with $n \leq m$, $k \leq n$ and $s \in \{1,\ 2,\ ...,\ m-n+1\}$, $f_k(s)$ represents the minimum cost to correspond points 1, 2, ..., *k* of contour 1, when point *k* has *s* matching hypotheses. With this formulation, we guarantee that the best global matching that preserves the absolute order is achieved.

To obtain the best global matching maintaining the relative order, it is necessary to rearrange the points of contour 2 (point 2 becomes point 1; point 3 becomes point 2 and so forth). Continuously, the correspondence of minimum cost that respects this new absolute order and the respective costs should be computed. The rearrangement process and consecutive calculus are repeated again, and so forth.

With the described approach, each new absolute order corresponds to a relative order, relatively to the initial arrangement. Thus, all of the possible relative arrangements of contour 2 are considered, being obtained all the matching that respect the relative order and the respective minimum costs.

In the example in study, it is necessary to solve 6 problems of global matching that respect the new successive absolute arrangements of the points of contour 2. After applying this formulation, the correspondence of minimum cost that respects the relative order of the points is again the previously presented.

4.2 Algorithm and implementation

Before we present our new algorithm, let us observe the example described in the previous section. In that example, we have, for instance:

$$f_3(3) = \min\{c_{33} + f_2(1), c_{34} + f_2(2), c_{35} + f_2(3)\}.$$

It seems that to calculate $f_3(3)$ we have to calculate three values and later compare them to choose the lower one. However, such is not necessary, because the values $c_{33} + f_2(1)$ and $c_{34} + f_2(2)$ were already calculated and $c_{34} + f_2(2) \leq c_{33} + f_2(1)$. According to this, it is enough to calculate $c_{35} + f_2(3)$ and compare it with $c_{34} + f_2(2)$. Thus, in each stage, only one sum operation and one comparison operation for each state is effectuated, if $s > 1$. If $s = 1$, then only one sum is done.

The algorithm presented starts from the hypothesis that is not known *a priori* a singular matching or a group of strong candidates to be matched. For that reason, it determines all the possible global matches that respect the new successive absolute orders and later chooses the one of minimum cost. The chosen matching is the one of lower cost that respects the relative order of the points.

Our new algorithm can be described as follows:
Algorithm:

1. Read the dimension of contours to be matched and the matching costs matrix *C*. Define the value of *n* and *m* so that $n \leq m$. If necessary ($n > m$), make the transpose of matrix *C*.

2. Repeat *m* times:

   (a) To $k = 1, 2, ..., n$ and $s = 1, 2, ..., m-n+1$, calculate the values of $f_k(s)$, taking in consideration what was referred before, avoiding repeated calculations already made. Keep the values of $f_k(s)$ in a table of *n* rows and $m - n + 1$ columns; that is, the used table must have so many rows as stages and so many columns as states, (Table 1).

(b) Determine and keep the minimum cost, which is the value kept in the position $(n,\ m-n+1)$ of the table of values. (In the previous example, it is the value kept in position $(4,\ 3)$ of Table 1).

(c) Define and keep the global correspondence of minimum cost, which is made by making a search and selection of cells in the built table. The selection of a certain cell $(i,\ j)$ means that the point *i* of contour 1 corresponds to point $i+j-1$ of contour 2. (See the cells used to define the matching in the example in study, Table 1.)

(d) Rearrange the columns of the matrix *C*; so that column 2 becomes column 1, column 3 becomes column 2 and so forth.

3. Seek for the minimum cost between the *m* kept values and the respective correspondence.

Table 1: Minimum costs kept by the algorithm for the example in study. The values are relative to the first problem (initial order) and the cells marked with * are used to define the correspondence.

	State (*s*)		
Stage (*k*)	1	2	3
1	$f_1(1)=1$	$f_1(2)=0$*	$f_1(3)=0$
2	$f_2(1)=4$	$f_2(2)=1$*	$f_2(3)=1$
3	$f_3(1)=6$	$f_3(2)=5$	$f_3(3)=1$*
4	$f_4(1)=11$	$f_4(2)=9$	$f_4(3)=2$*

If one singular matching is known *a priori*, then the algorithm does not need to determine all the possible global matching as in the presented case. For instance, let us suppose that it is known that point *i* of contour 1 should correspond to point *j* of contour 2. Then, the points of both contours are rearranged: point *i* of contour 1 becomes point 1, point $i+1$ becomes point 2 and so forth. The same is made in contour 2. Now it is enough to solve one single problem to determine the best global matching that respects the new absolute order, instead of *m* problems that the algorithm will have to solve if no singular correspondence was known.

4.3 Computational cost

Considering a contour defined by *n* points and the other defined by *m* points, with $n \leq m$, for each global matching that respect the absolute order there are *n* stages and $m-n+1$ states. For each stage, only one sum for state is effectuated. For each state larger than 1 only one comparison is effectuated. Thus, we have, in the total, $n \times (m-n+1)$ sums and $n \times (m-n)$ comparisons, only counting the fundamental operations.

To obtain the best global matching respecting the relative order, we have to solve *m* problems; therefore, there are $m \times n \times (m-n+1)$ sums and $m \times n \times (m-n)$ comparisons. To choose the best global correspondence from among all the global ones, we have more $m-1$ comparisons. Thus, the computational complexity is $O(m \times n \times (m-n+1))$.

From the exposed, we can conclude that the time of execution will grow when the number of points that defines the contours grows and decreases when the difference among the number of points of the two contours decreases too.

5 Dynamic programming with restriction of order versus Hungarian Method, Simplex for Flow Problems and LAPm

5.1 Test conditions

Before presenting some of the experimental results obtained, it is important to refer that this comparison was accomplished after the implementation of our new algorithm of dynamic programming in the computational platform for image processing and analysis already referred, [20], [8]. To compare the optimization methods – assignment algorithms without order restriction (AAWOR) and the dynamic programming algorithm with order restriction (DPAWOR) – it was considered affinity matrices obtained using a methodology integrated in the same platform. This methodology is based on geometric modeling and modal matching, as proposed by Shapiro, [18], [20], [2].

To compare the optimization algorithms based on the Hungarian method, Simplex for Flow Problems and LAPm with the new optimization algorithm based on dynamic programming, it is fundamental that the process to determine the cost matrix associated to the points that define both contours is exactly the same. Thus, in all of the experimental tests done, we accepted the configuration defined by default, in the computational platform used, for the building process of the affinity matrices.

In the definitions of the Simplex for Flow Problems algorithm already integrated in the computational platform adopted, the configuration defined by default was also used, because it is, in general, the fastest, Figure 4. To get the time spent by each one of the optimization algorithms considered, a function already available for that purpose in the same platform was used.

5.2 Results

The quality of the matches obtained using AAWOR and DPAWOR algorithms, in most of the contours tested, weas exactly the same and of high quality. The differences appeared when AAWOR present crossed matches, what obviously does not happen with DPAWOR.

To illustrate the differences of the matches found by the two types of algorithms considered in some experimental cases, observe the Figures 2, 5, 6, 7, 8, 9 and 10. In those figures, the contours were aligned using the rigid transformation estimated from the matches found, using unitary *quaternions*, [20]. In some of the same figures, there are small differences in the positions of the contours, because the angle of rotation of a contour in relation to the other one is obtained after the matching process. Thus, bad matches can originate an incorrect estimation for the rotation angle involved.

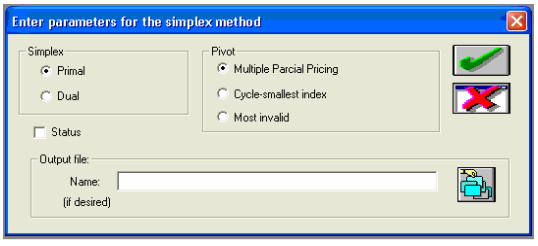

Figure 4: Configuration defined by default in the computational platform used for the optimization algorithm based on the Simplex algorithm.

Figure 5: Matching of the contours of Figures 1 and 2 using the dynamic programming algorithm.

In Table 2, we present the computational times needed to determine the matching of several pairs of ordered contours and the respective matching costs. Some of the matching indicated are not illustrated in this paper because they were equal for the two types of algorithms in comparison, or present almost imperceptive differences (as was the case of the matching between contours "*foot13*" and "*foot2*"). It is important to refer that the cost of the global matching depends on the elements of the matching cost matrix and this one depends on the contours and the values of the parameters considered in the Shapiro's matching methodology. The time indicated is an average time, because small variations were observed.

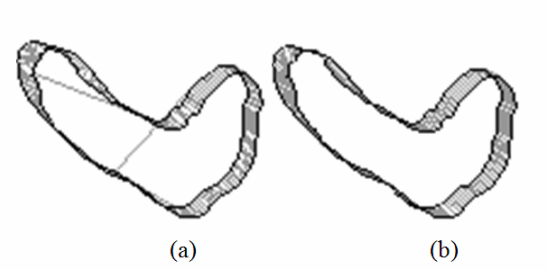

Figure 6: Matching of contours "*foot13*" and "*foot14*", (pedobarography images, [7]) defined by 233 and 253 points, respectively: (a) matching using AAWOR, (b) matching using DPAWOR.

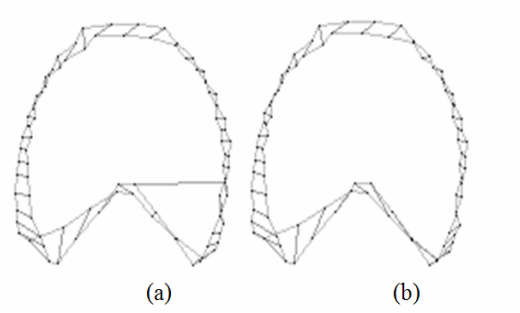

Figure 7: Matching of contours "*rib1*" and "*rib2*", defined by 46 points each: (a) matching using AAWOR, (b) matching using DPAWOR.

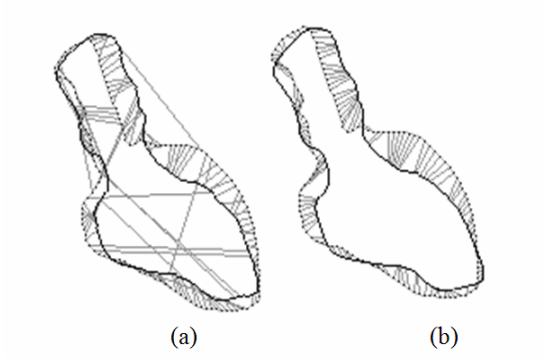

Figure 8: Matching of contours "*heartB3*" and "*heartB2*", defined by 389 and 139 points, respectively: (a) matching using AAWOR, (b) matching using DPAWOR.

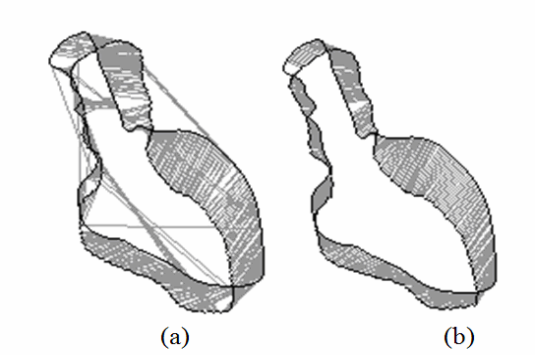

Figure 9: Matching of contours "*heartB3*" and "*heartB4*", defined by 389 and 417 points, respectively: (a) matching using AAWOR, (b) matching using DPAWOR.

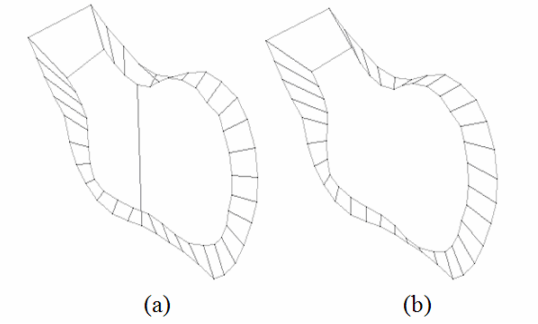

Figure 10: Matching of contours "*heartB3*" and "*heartB4*", defined by 389 and 417 points, respectively: (a) matching using AAWOR, (b) matching using DPAWOR.

Table 2: Comparison between AAWOR and DPAWOR algorithms. (The experimental tests were done using the referred computational platform, running in a PC Pentium III, at 1GHz, with 256MB of RAM and Microsoft Windows XP.

Number of points of the contours		Execution times [s]			
		Hungarian	*Simplex*	*LAPm*	Dynamic
28	28	4.29	0.02	0.01	<0.01
36	36	>60	0.04	2.36	<0.01
46	46	>60	0.06	2.77	<0.01
86	57	>60	0.20	1.33	0.01
81	84	>60	0.20	2.43	<0.01
233	67	>60	1.33	15.98	0.25
233	253	>60	2.01	>60	0.15
389	139	>60	5.42	>60	3.80
389	417	>60	9.86	>60	1.19

6 Conclusions and future work perspectives

Relatively to the quality of the matches found, the AAWOR algorithms, obviously, always present a matching of minimum cost, because they are subjected to the same restrictions. As only in very singular situations, more than one matching of minimum cost exists, the matching obtained by the three assignment algorithms was always the same one.

The comparison between the results obtained using AAWOR algorithms and DPAWOR algorithm allows us to conclude the following:

1. Whenever the AAWOR reached a good matching without crossed matches, the DPAWOR reached the same matching; therefore, the global cost of the matching was exactly the same for the two types of algorithms.

2. When the AAWOR reached a matching with some crossed matches, the DPAWOR reached an identical matching but without crossed matches. Obviously, the cost associated was superior because the restriction of the order forced some crossed matches to be substituted by matches of higher costs but more coherent.

3. In the situations where the matching obtained by AAWOR was senseless, then the matching obtained using DPAWOR also was. It is important to refer that those bad matching is not due to the optimization algorithms adopted but to the methodology used in the construction of the matching cost matrix. Thus, any example of this situation was not presented in this paper.

The execution time of the DPAWOR algorithm was always inferior to the execution time of all the AAWOR algorithms, independently of the contours having been defined for equal or different number of points, or composed by many points or just by few points. As it can be observed in Table 2, there were situations in that the computational platform indicated execution times around 0 (zero) seconds for DPAWOR, meaning very low computation times. Besides, the computer used is very slow, when compared with the modern computers.

It can be verified that the execution times of the DPAWOR algorithm varied in agreement with what was anticipated in section 4.3. In other words, the time increased when the number of points that defines the contours increased, but it decreased when the difference between the number of points that defines the two contours vanished.

Finally, as perspectives of future work, we hope to apply our DPAWOR algorithm to determine the matching of characteristic points of objects represented in images using several methodologies for the determination of the matching cost matrix, where the order of the

points, or other characteristics of the shapes or images involved should be considered and preserved.

Acknowledgements

This work was partially done in the scope of project "Segmentation, Tracking and Motion Analysis of Deformable (2D/3D) Objects using Physical Principles", financially supported by FCT - Fundação para a Ciência e a Tecnologia from Portugal, with reference POSC/EEA-SRI/55386/2004.

References

[1] Amico, M. D. and Tooth, P. Algorithms and codes ford dense assignment problems: the state of the art. *Discrete Applied Mathematics*, 100:274–278, 2000.

[2] Bastos, L. Optimização da determinação das correspondências entre objectos deformáveis no espaço modal. Master's thesis, Faculdade de Engenharia da Universidade do Porto, Portugal, 2003.

[3] Bastos, L. and Tavares, J. Matching of objects nodal points improvement using optimization. *Inverse Problems in Science and Engineering*, 14(5):529–541, 2006.

[4] Carcassoni, M. and Hancock, E. Correspondence matching with modal clusters. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 5(12):1609–1615, December 2003.

[5] Fielding, G. and Kam, M. Weighted matching for dense stereo correspondence. *Pattern Recognition*, 33(9):1511–1524, 2000.

[6] Hillier, F. and Lieberman, G. *Introduction to operations research*. McGraw-Hill International Editions, 1995.

[7] J. Tavares, J. B. and Padilha, J. Matching image objects in dynamic pedobarography. In *RecPad 2000 – 11th Portuguese Conference on Pattern Recognition*, Portugal, 2000.

[8] J. Tavares, J. B. and Padilha, J. Apresentação de um banco de desenvolvimento e ensaio para objectos deformáveis. *RESI - Revista Electrónica de Sistemas de Informação*, 1(1), 2002.

[9] Latecki, L. and Lakämper, R. Shape similarity measure based on correspondence of visual parts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(10), October 2000.

[10] Löbel, A. Mcf – a network simplex implementation. http://www.zib.de/optimization/software/mcf, 2000.

[11] Maciel, J. *Global matching: Optimal solution to correspondence problems*. PhD thesis, Instituto Superior Técnico, Lisboa, Portugal, 2001.

[12] Norman, J. *Elementary dynamic programming*. Eduard Arnold, London, 1975.

[13] Oliveira, F. and Tavares, J. Matching contours in images using curvature information. In *VIPimage – I ECCOMAS Thematic Conference on Computational Vision and Medical Image Processing*, pages 375–379. Taylor & Francis, 2007.

[14] S. Belongie, J. P., J. Malik. Shape matching and object recognition using shape contexts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(24), April 2002.

[15] S. Caetano, D. S., T. Caelli and Barone, D. Graphical models and point pattern matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(10), October 2006.

[16] Sclaroff, S. and Pentland, A. Modal matching for correspondence and recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(6), 1995.

[17] Scott, C. and Nowak, R. Robust contour matching via the order-preserving assignment problem. *IEEE Transactions on Image Processing*, 15(7):1831–1838, July 2006.

[18] Shapiro, L. and Brady, M. Feature-based correspondence: an eigenvector approach. *Image and Vision Computing*, 10(5):283–288, 1992.

[19] Starink, J. and Backer, E. Finding point correspondences using simulated annealing. *Pattern Recognition*, 28(2):231–240, 1995.

[20] Tavares, J. *Análise de movimento de corpos deformáveis usando visão computacional*. PhD thesis, Faculdade de Engenharia da Universidade do Porto, Portugal, 2000.

[21] Volgenant, A. Linear and semi-assignment problems: a core oriented approach. *Computers and Operations Research*, 23(10), 1996.

[22] Winston, W. *Operations research: applications and algorithms*. Duxbury Press, USA, 3rd edition, 1994.