STRUCTURE DESCRIPTION AND MATCHING

J. Rošický

Czech Technical University in Prague, Faculty of Mechanical Engineering, Department of Instrumentation and Control Engineering

Abstract

Structure description and matching are useful techniques with wide area of applications. Common motivation could be the need for object identification. This work presents a specific method developed to describe and match particular type of structures – sets of points in two-dimensional space. The method incorporates creation of the structure description suitable for further matching and a procedure for statistical evaluation of the match of two structures. Some details regarding the implementation in MATLAB are presented as well.

1 Introduction

The motivation for development of a method for structure description and matching comes from the field of physical object identification. There are many professions interested in such a task. For example merchants, transport and logistics services providers, accountants, state institutions, libraries, archives, museums, research institutions, security services, even criminalists, just to name a few. People from mentioned professions deal with diverse objects (artifacts), but their common question is: "What information is associated with the given object?" A very common example of "an object" identification task is the finger-print recognition. Given a *finger* and a *register* of scanned finger-prints a criminalist could ask "Is the finger-print of the given finger stored in our register?" Methods for physical object identification usually share general approach depicted in Fig. 1. If objects can be considered as sets of points in two-dimensional space then "Object *representation* (abstract *structure*)" in Figure 1c) is a set of pairs of two-dimensional coordinates.

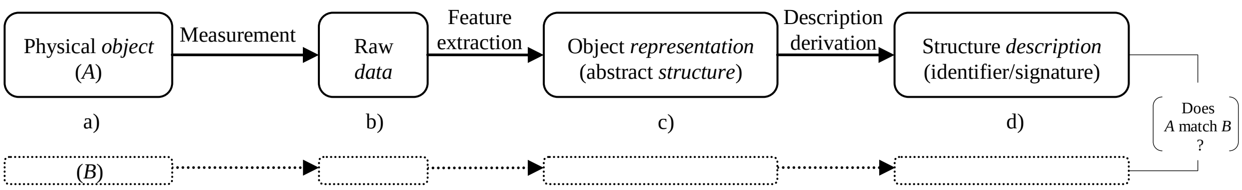

Figure 1: Identifying an object.

2 Matching two sets of points

The task formulation comes out from c) in Figure 1. Given two sets of points in 2D space (two object *representations*, Figure 2 and Figure 3) we want to construct an indicator of the quality of their match.

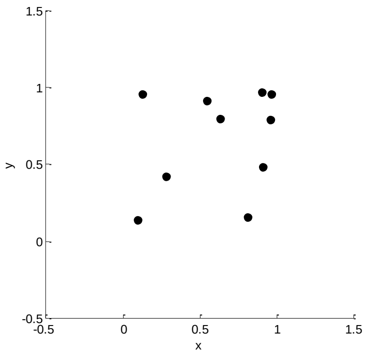

Figure 2: Representation of an object *A*.

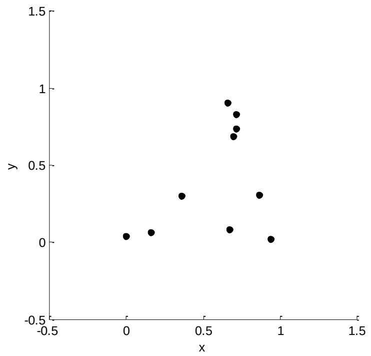

Figure 3: Representation of an object *B*.

The positions of points in sets representing a single object can be transformed – translated, rotated (Figure 4), perturbed by noise (Figure 6) and some points could be missing (Figure 5). We do not consider the change of the scale.

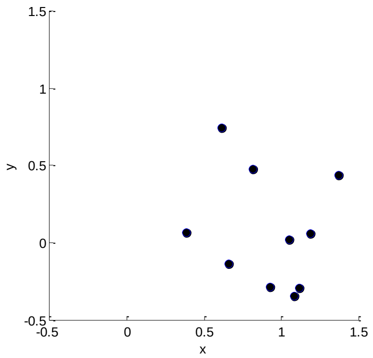

Figure 4: Representation of the object *A*, translated and rotated.

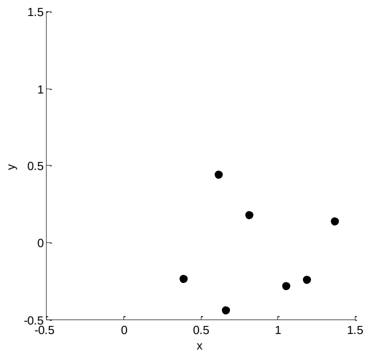

Figure 5: Representation of the object *A*, translated, rotated and 3 points are missing.

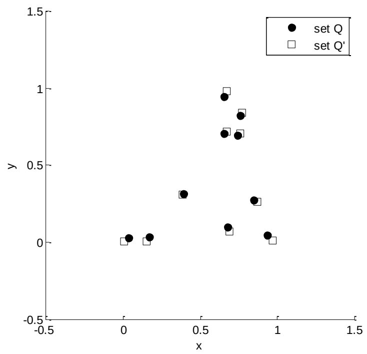

Figure 6: Two representations of a single object *B*. Points are perturbed by noise.

3 A brief review of approaches

Number of methods was developed to solve the task of matching two sets of points. The procrustes algorithm implemented in Matlab [2] determines a linear transformation of the points in one set to best conform them to the points in the second set. However the sets must have the same number of points, which disqualifies this method according to our task formulation. The approach which comes from the field of analyzing protein spot patterns within two-dimensional electrophoresis [3] is rather linked/associated/connected with the protein spot application and thus not so general as [XXX] or [1]. The Murtagh's algorithm [1] with the concept of the "world-view vector" is very close to our approach. However we do not try to determine the transformation between the two matched point sets and we use a different approach to evaluate the match.

4 The Proposed Solution

We describe the given sets in terms of sets of invariant features according to translation, rotation and loss of points. We do not consider scaling. Then we construct the feature space of gathered features. Finally we analyze the feature space to estimate if there is an indication of a match of the given sets of points.

The description of a set of points is constructed as a set of pairs of the length d of the connecting line between each pair of points and the orientation φ (angle) of the connecting line according to the horizontal axis. An example of such a description is depicted in Figure 7 for a set of four points (features of connecting lines 1-2 and 3-4 are omitted for brevity). The corresponding feature space is depicted in Figure 8.

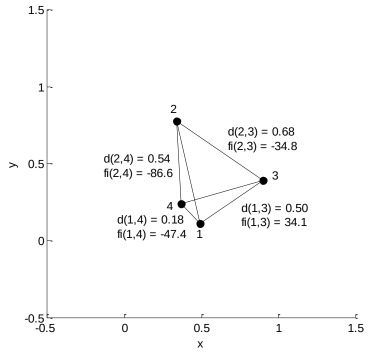

Figure 7: Description of a set of points – lengths of connecting lines *d* and their orientations *φ* [deg].

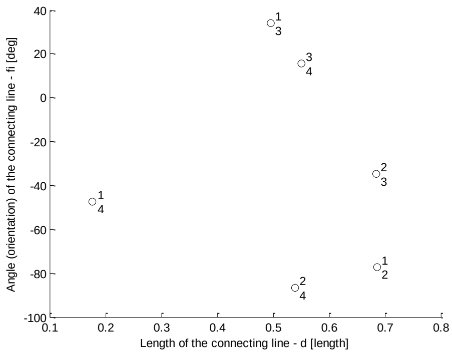

Figure 8: Feature space of lengths *d* and angles *φ* of the set of points from Figure 7.

For two sample sets of points we have the corresponding figures – Figure 9 and Figure 10.

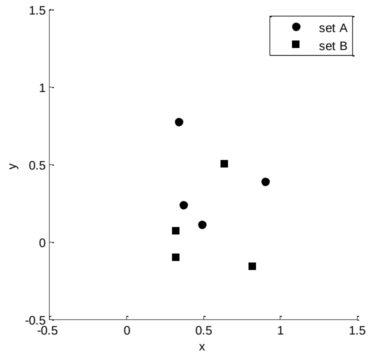

Figure 9: Two sets of points.

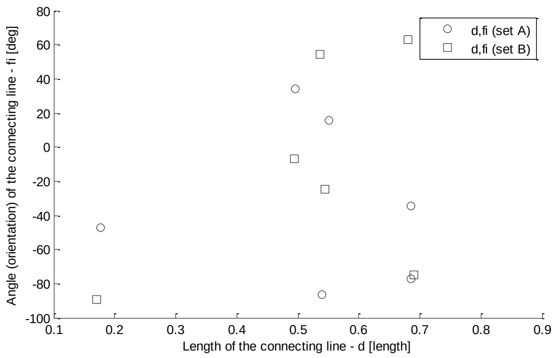

Figure 10: Feature space with *set A*'s and *set B*'s features.

Now we can analyze the feature space. We will create a set of angle differences Φ: for each *set A*'s feature (d_A,φ_A) we find all *set B*'s features (d_B,φ_B) with $|d_A-d_B| \leq \delta$ and compute the angle differences φ_A - φ_A in the range [0°, 180°]. We plot the distribution of the angle differences from Φ and evaluate the similarity of the two sets. The presence of a *peak value* in the distribution (Figure 11) indicates that the two sets *do match*; the "flatness" of the distribution (Figure 12) indicates that given distributions *do not match*. The quantity *match ratio* indicating the *do/do not match* result is defined as the ratio of the peak (maximum) value and the mean of the rest values in the distribution of the set Φ.

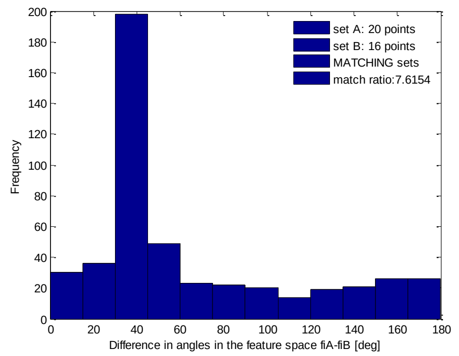

Figure 11: Distribution of the angle differences. The *peak value* indicates that the two sets *do* match.

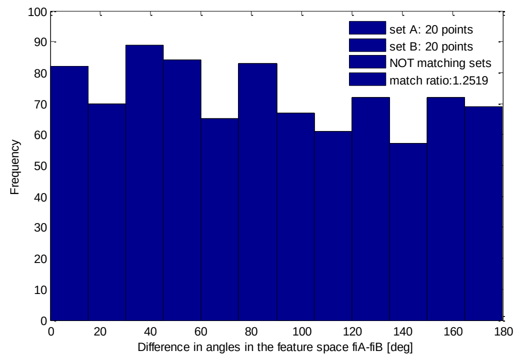

Figure 12: Distribution of the angle differences in the feature space. The "*flatness*" of the distribution indicates that the two sets *do not* match.

5 Implementation (and Experiments)

The proposed method for matching two sets of points was implemented in Matlab. Some useful Matlab functions used are briefly noted.

A set of 2D points *A* was generated using `rand`:

```
N = 20; % number of points
A = rand(N,2);
```

The lengths of connecting lines of points from *A* were calculated using pair wise distance function `pdist`:

```
d = pdist(A); % vector of lengths of connecting lines
```

Sometimes it was useful to work with the matrix form of *d*:

```
D = squareform(d) % the matrix form of d
```

The orientations φ (angles) of the connecting lines were computed using pair wise distance function `pdist` together with the metric function `distfun`:

```
distfun = @(u,V) atand((V(:,2)-u(1,2))./(V(:,1)-u(1,1)));
fi = pdist(A,distfun); % vector of angles of the connecting lines
```

To quantify the level of the match of two given sets of points the quantity `qmatch` (*match ratio* in Figure 11 and Figure 12) is computed as the ratio:

$$\text{qmatch} = \frac{\text{maximum value}}{\text{mean of the rest values}} \tag{1}$$

(the maximum value - the mean of the rest values) / standard deviation of the rest values of the distribution of the set of angle differences Φ (implemented as `dFIg`):

```
[freq,xout] = hist(dFIg); % Create the array of frequencies freq
fmax = max(freq);
qmatch = fmax / mean(freq(freq<fmax));
```

5.1 Performance - time

Table 1 shows the results of three time performance tests. In each test two sets were matched. The sets had the same number of points and were *matching* sets – i.e. one set was a transformation of the other set (rotated, translated and perturbed by noise). The magnitude of noise was set to 1% in each coordinate. The tests were executed on an IBM T60 notebook (2CPUs, 1.8GHz, 2.5GB RAM) with MATLAB R2009a.

Number of points in each set	Matching time [sec]
10	~ 0.02
20	~ 0.1
50	~ 2.5

Table 1. Performace tests - time.

The results indicate that further optimization is desirable for potential deployment of the method in a real-time application where a quick response (under 0.1 seconds) might be mandatory.

5.2 Performance – loss of points

Table 2 shows the *qmatch* quantity defined in (1) indicating the performance of the method with regard to the proportion of loss of points. Three groups of tests were executed. In each test pair of *matching* sets (see 5.1 Performance – time) was evaluated 20 times for four conditions (0, 10, 20 and 50%) of loss of points. Table 3 shows the results for the same test except the sets were *not matching*. The magnitude of noise was set to 1% in each coordinate.

qmatch (& its std. deviation)	Loss of points in the second (*matching*) set			
Number of points in the first set	0%	10%	20%	50%
10	15.5 (3.4)	18.0 (4.3)	14.9 (3.4)	18.4 (7.6)
20	4.0 (0.4)	3.9 (0.4)	4.0 (0.4)	3.8 (0.5)
50	1.5 (0.05)	1.5 (0.06)	1.5 (0.6)	1.5 (0.09)

Table 2. The quantity *qmatch* (and its standard deviation) for various proportions of loss of points. Matching sets, number of measurements: 20.

qmatch (& its std. deviation)	Loss of points in the second (*not matching*) set			
Number of points in the first set	0%	10%	20%	50%
10	2.1 (0.4)	2.1 (0.4)	2.1 (0.3)	2.3 (0.5)
20	1.25 (0.06)	1.3 (0.1)	1.3 (0.2)	1.5 (0.2)
50	1.05 (0.02)	1.05 (0.02)	1.06 (0.03)	1.1 (0.05)

Table 3. The quantity *qmatch* (and its standard deviation) for various proportions of loss of points. Not matching sets, number of measurements:20.

5.3 Performance – noise perturbation

Table 4 shows the *qmatch* quantity defined in (1) indicating the performance of the method with regard to the proportion of noise magnitude in coordinates. Noise magnitude 1% means that the second (*matching*) set of points was generated from the first set by changing the position of each point +/-1% in each coordinate. Table 5 shows the same experiment for *not matching* sets.

qmatch (& its std. deviation)	Noise magnitude			
Number of points in each set	1%	2%	5%	10%
10	**	7.2 (2.6)	2.8 (0.6)	1.8 (0.4)
20	**	2.2 (0.3)	1.6 (0.2)	1.3 (0.2)
50	**	1.4 (0.07)	1.2 (0.05)	N/A*

*) Computational time exceeded 60 seconds.
**) The same values as in Table 2 for 0% loss of points are assumed.

Table 4. The quantity *qmatch* (and its std. deviation) for various magnitudes of noise perturbation. *Matching sets*, number of measurements: 20.

qmatch (& its std. deviation)	**Noise magnitude**			
Number of points in each set	1%	2%	5%	10%
10	**	1.5 (0.3)	1.4 (0.2)	1.27 (0.08)
20	**	1.20 (0.09)	1.14 (0.08)	1.11 (0.05)
50	**	1.04 (0.02)	N/A*	N/A*

*) Computational time exceeded 60 seconds.
**) The same values as in Table 3 for 0% loss of points are assumed.

Table 5. The quantity *qmatch* (and its standard deviation) for various magnitudes of noise perturbation. *Not matching* sets, number of measurements: 20.

6 Conclusion

A method for matching two sets of points was designed. The method is based on the concept of the "world-view vector" proposed in [1]. Three tests were executed to estimate the performance of the method regarding computational time, loss of points and noise perturbation. For sets with more than 20 points further optimization would be necessary in the case of a requirement for fast matching times (<0.1 sec). The results of the loss-of-points-tests indicate that the method have a potential for robustness in this regard. The noise perturbation has significant effect regarding the robustness of the method. With the higher noise magnitudes the discriminative ability of the method drops down.

The future work will be focused on several areas. Some kind of weighting of the features might improve overall performance. The analysis of the features in the feature space is time demanding as the number of points increases. One possible approach could be use of convolution instead of the "delta-neighborhood" scanning. The *qmatch* indicator is sensitive to the choice of intervals for analyzing the distribution of the set Φ. It could happen that the peak frequency splits in two neighboring intervals and then the *qmatch* is distorted. The next work will be aimed to find more suitable indicator. The future *qmatch'* quantity should also reflect the difference in the number of points in the two matched sets.

References

[1] Murtagh F.: *A New Approach to Point Pattern Matching*. Publications of the Astronomical Society of the Pacific, 1992, in press.

[2] Mathworks: *Procrustes*. Matlab Documentation. Available on-line: [http://www.mathworks.com/access/helpdesk/help/toolbox/stats/procrustes.html]

[3] Klaus Kriegel et al.: An alternative approach to deal with geometric uncertainties in computer analysis of two-dimensional electrophoresis gels. Electrophoresis 2000, 21, 2637-2640