An exhaustive comparison framework for distributed shape differentiation in a MEMS sensor actuator array

Kahina Boutoustous Eugen Dedu
Julien Bourgeois
Laboratoire d'Informatique de l'Université de Franche-Comté (LIFC)
1 cours Leprince-Ringuet, 25200 Montbéliard, France
{Kahina.Boutoustous, Eugen.Dedu, Julien.Bourgeois}@pu-pm.univ-fcomte.fr

Abstract

The Smart Surface [1] project aims at designing an integrated micro-manipulator based on an array of micromodules connected with a 2D array topology network. Each micromodule comprises a sensor, an actuator and a processing unit. One of the aim of the processing unit is to recognize the shape of the part that is put on top of the smart surface. This recognition or more precisely this differentiation is done through a distributed algorithm that we call a criterion. The aim of this article is to present the ECO framework, which is able to test exhaustively the efficiency of different differentiation criteria, in term of differentiation efficiency, memory and processing power needed. The tests will show that ECO is of great help for choosing the best criteria to implement inside our smart surface.

Keywords: *shape differentiation, distributed computing, MEMS.*

1 Introduction

During an assembly process, it is necessary to feed assembly line workstations with well-oriented and well-positioned parts. These parts are often jumbled and they need to be sorted and conveyed to the right workstation. To do so, the operations to be performed on parts are the following: identifying, sorting, orienting, positioning, feeding, and assembling. Among the most promising solutions to perform these tasks on microparts, is the combination of micro-electro mechanical systems (MEMS) in order to form an actuator arrays. However, if a single microactuator is not powerful enough to move a micropart, several microactuators working cooperatively might very well do it. A MEMS sensor/actuator arrays with embedded intelligence is referred as a smart surface.

The objective of the Smart Surface project is to design such an integrated MEMS system which will be able to identify, to sort, to orient and position microparts. This article deals only with the identification part of the process: A micropart is put on the Smart Surface which have to recognize the part shape and give the proper orders to the control system to move it on the right place. In fact, recognition is not the proper term. Given a set of part, the Smart Surface have to *differentiate* all the parts within the set. As the processing power of the Smart Surface is embedded in very limited space, this differentiation process has to be optimized both in term of memory used and processing power needed. The differentiation is made by a distributed program loaded in the Smart Surface. For the rest of the paper we call this program a differentiation *criterion*. The aim of the Exhaustive COmparison (ECO) framework which is presented in this article is to test exhaustively, i.e. for all kinds of possible part shapes, a set of criteria to choose the most adapted ones. The main condition for choosing a criterion is that it must be able to differentiate all the possible parts, that is what we call total differentiation. The other two remaining conditions are: using the less memory and using the less computing power.

The rest of the paper is organized as follows. Section 2 details the Smart Surface project. Section 3 presents the ECO framework, while the tests are performed on section 4. Some related works to shape representation are presented section 5 and they are followed by the conclusion and presentation of future works.

2 The Smart Surface Project

There have been numerous projects of MEMS actuator arrays in the past and more precisely in the 1990's.

[1]This work is funded by the French National Agency for Research, by the Doubs departemental council and by the University of Franche-Comté.

These pioneer researches have developed different types of MEMS actuator arrays, based on actuators either pneumatic [18, 9], servoed roller wheels [14, 15], magnetic [12] or thermobimoph and electrostatic [22]. Some of these preliminary studies use a sensorless manipulation scheme based on the Goldberg's algorithm [11] for parallel jaw grippers. The jaw grippers are obtained with MEMS actuator arrays by creating opposite field forces which then can orient and move the parts. Bohringer et al. [2] have proposed a concept called "programmable force field" which is an extension of the Goldberg's algorithm. This manipulation scheme which is well-adapted for jaw grippers has shown some limitations when adapted to MEMS actuator arrays. For instance, the absence of a command law can lead to uncertain behaviours [17] or MEMS actuator arrays has to be programmed for each different kind of parts. More recent research has been conducted in order to include sensors and to add intelligence to MEMS actuator arrays but it either fails to develop it at a micro-scale [1] or to be fully integrated [10].

The objective of the Smart Surface project is to design a distributed and integrated micro-manipulator based on an array of micro-modules in order to realize an *automated positioning and conveying surface*. Each micro-module will be composed of a micro-actuator, a micro-sensor and a processing unit. The cooperation of these micro-modules thanks to an integrated network will allow to recognize the parts and to control micro-actuators in order to move and position accurately the parts on the smart surface. The parts are small, they cover a few numbers of micro-modules (e.g. 4×4).

Figure 1 shows the Smart Surface. The rectangular holes seen on the front-side are the air nozzles. Air-flow comes through a micro-valve in the back-side of the device and then passes through the nozzle. The advantage of this solution is the micro-actuators, the most fragile part of the surface, that are protected. The circle holes are used by the micro-sensor to detect the presence or not of the part on the surface.

The strength of our project is the multidisciplinary collaboration between six labs specialized in their field and more than twenty researchers. We are responsible for the information management inside the smart surface, i.e. distributed differentiation of the part and communication infrastructure.

.

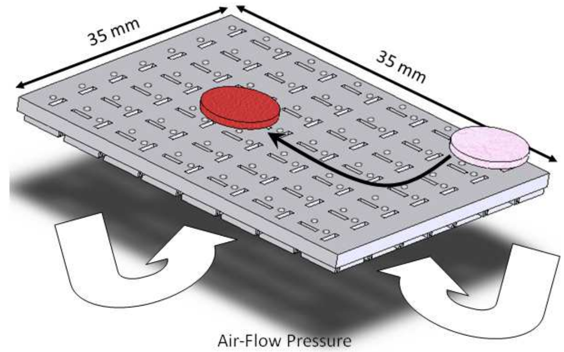

Figure 1. An overview of our smart surface.

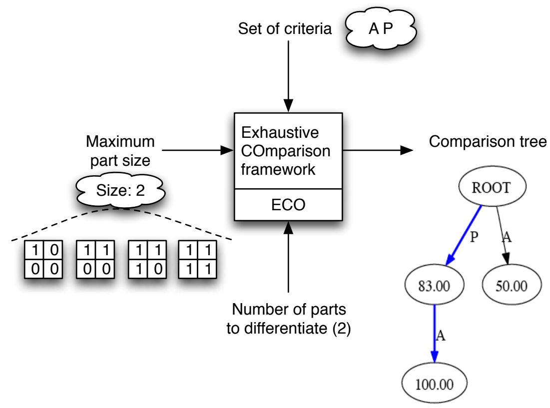

Figure 2. Overview of the framework.

3 The ECO framework

Before implementing the part differentiation algorithms on the Smart Surface, we are interested to find out criteria allowing high differentiates.

This section presents a framework for criteria comparison in differentiating parts, based on an exhaustive part generation. The framework is presented in figure 2. It receives as input a set of criteria, the maximum part size (a square) and the *number* of parts to differentiate. The framework exhaustively generates all the appropriate parts. It generates several comparison trees: differentiation tree, cost tree. An example of question which the framework answers to is: What criteria differentiate best three random parts not greater than 3×3? The constraints of the framework which will be relaxed in future works are :

- Parts can be rotated only at $90°$.
- No error in sensors and communications.
- We work on *family of parts*. We define a family of parts all the *ideal* parts which have the same image (discrete representation) on the surface. For example, the typographic letter L and L (with and sans serifs) have the same image on the surface, because the serifs are much smaller than the sensors.

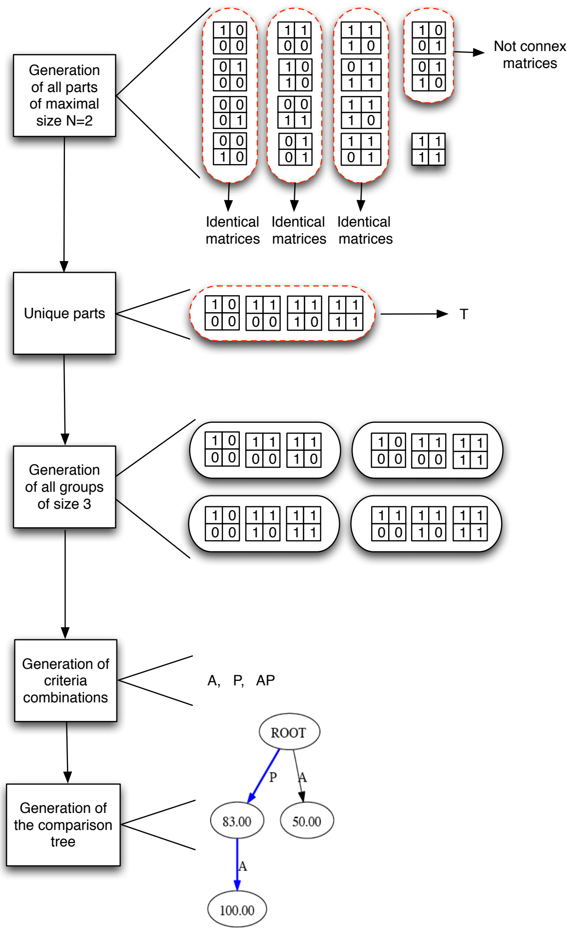

Figure 3. Global structure of our model.

The parts on the Smart Surface are supposed to be represented by square matrices of size 3 or 4. In order to find criteria reaching 100% differentiation, all possible parts of size $P \times P$ with $P = 3$ and $P = 4$ are considered. This set of parts is used to generate groups of parts. These groups are used to test the criteria or combinations of criteria which reach total differentiation. Our method is divided into five steps (see fig. 3):

- First, all the parts of size $P \times P$ are generated.
- Afterwards, the resulting set of parts is reduced by eliminating translations, 90° rotations and mirrors (see tab. 1), as detailed in Section 3.1.
- Afterwards, all the combinations of n parts from the previously generated parts are generated (see tab. 1).
- Afterwards, all the combinations of CC_i criteria are generated. For example, if $C_i = \{A, S, P\}$ is the set of criteria, the generated combinations are $CC_i = \{\{A\}, \{S\}, \{P\}, \{AS\}, \{AP\}, \{SP\}, \{ASP\}\}$. This means that all the criteria are combined in order to differentiate the parts. Several criteria have been tested, presented in section 4.
- Finally, there is the differentiation phase, detailed in section 3.2.

Table 1. The number of unique parts and the number of generated groups of three parts.

Max part size	Number of parts generated	Number of unique parts (T)	Number of groups
3×3	512	35	$C^3_{35} = 6545$
4×4	65536	1280	$C^3_{1280} = 348706560$

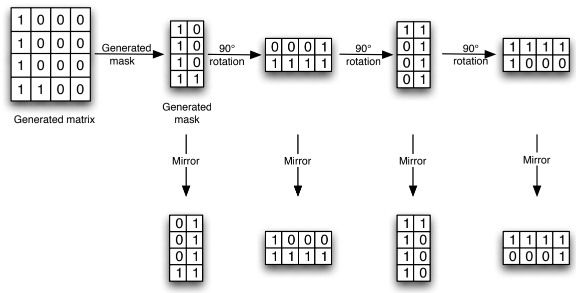

Figure 4. Part generation.

3.1 Generation of the parts

Figure 4 presents the steps used in the generation of all the parts. A part on the $P \times P$ square may be represented as a binary matrix, as shown in figure 5. In a $P \times P$ square, there are $2^{P \times P}$ parts. However, many of them are not *connex*, i.e. in fact there are two parts instead of one. The connexity checking is done with a research in depth.

Afterwards, its *mask* is generated. The mask is a matrix generated from the initial matrix where the first columns and first lines with only 0s are removed (see figure 5). This step remove translated identical parts.

During the next step, masks are rotated 90°, 180° and 270°, each mask is mirrored. After each 2 by 2 comparison, identical masks are removed such that only one mask of same type remains.

The parts remaining after this process are unique compared to translation, rotation of multiple of 90° and mirroring. Let T be the total number of unique parts (see tab. 1).

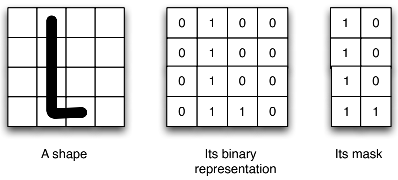

Figure 5. Part discretization.

3.2 Differentiation of the parts

For each criterion C_j and each group G_i of parts a differentiation matrix D is generated, with $D(i,j) = 1$, if the values of the criterion between the two parts i and j are identical, otherwise it is 0.

$$\begin{array}{c} D_{G_i,C_j}(k,l) = 1, \forall k,l \in P \\ \Leftrightarrow C_j \text{ differentiates all } P \in G_i \end{array} \tag{1}$$

In the case of a combination of several criteria CC_j the union of the differentiation matrices is computed. If the matrix D contains only 1 values, the parts are said to be differentiated according to this combination of criteria.

$$\begin{array}{c} D_{G_i,CC_j} = \cup_{k \in CC_j} D_{G_i,C_k} \\ CC_j \subset \{C_1, C_2, ..., C_m\} \end{array} \tag{2}$$

The matrix *D* is upper triangular. A differentiation is said to be total if the matrices are differentiated according to *all* possible groups.

Fig. 6 is an example of the computation of a differentiation matrix with $G_1 = \{P_1, P_2, P_3\}$ for the combination of criteria $CC_1 = \{ASP\}$.

The major challenge is to find out a combination of criteria which leads to a total differentiation, i.e. for any group of parts, a differentiation of the parts using one combination of our criteria is always achieved.

The following algorithm details this process: Let n be the number of parts which are needed to be differentiated. The framework generates all the groups of n parts. If T is the total number of unique parts, there will be C_T^n groups. The algorithm is:

```
for each CC_i = subset of {C_1, C_2, ..., C_m} do
  for each group G_i subset of n elements in P do
    if CC_i is a criterion then
      build the differentiation matrix D_{G_i,CC_i}
    else
      {CC_i is a combination of criteria}
      for each C_j in CC_i do
        build the differentiation matrix D_{G_i,CC_j} = ⋃ D_{G_i,C_j}
      end for
    end if
    compute the differentiation rate t
  end for
  compute the average ta of all differentiation rates t
end for
build comparison tree
```

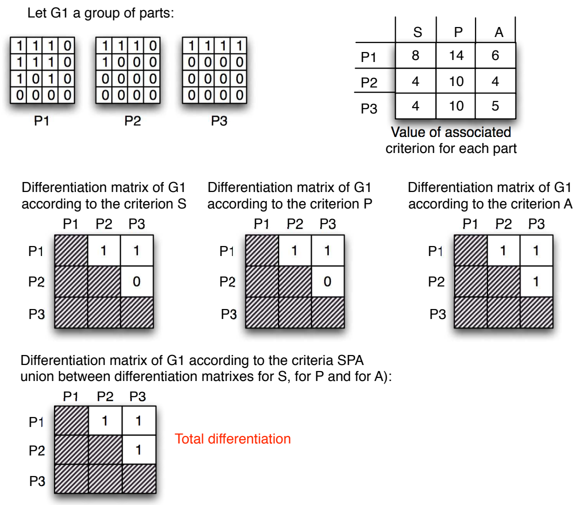

Figure 6. An example of a group differentiation according to a combination of criteria.

The usefulness of criteria is presented as a simplified tree (path XY is the same as YX) called *comparison tree* (see fig. 9 in section 4.3). Each node has a value expressed as percentage of differentiation using all the criteria of the path from the root of the tree. Finally, a cost (execution time, memory used etc.) is associated to each branch.

3.3 Memory costs

The smart surface has memory limited due to is micro-scale integration. This section formalizes the memory needed by all criteria.

First, the memory needed by one cell (micro-module) using one criterion is computed. Each cell knows all the models. Let n be the number of models (the parts in the set which should be differentiated), e.g. $n = 3$. Let $m(C_i, P_j)$ be the memory needed (in bits) to store the value of the criterion C_i of the model part P_j. Therefore, all the models need M_1 bits memory, with $M_1 = \sum_{j=1}^{N} m(C_i, P_j)$. When a part P is on the smart surface, it is first constructed by each cell. It occupies $M_2 = P \times P$ bits, because the maximum size of a part is P by P cells. The value of a criterion of the part P is $M_3 = m(C_i, P)$. Section 4.3 gives some practical values for M_3. The sum of these values gives the total memory needed by one cell for one criterion C_i:

$$m = M_1 + M_2 + M_3 \tag{3}$$

Let CC_j be the set of combinations of criteria that conduct to a total differentiation. The best set from memory point of view is:

$$M = \{c_k / c_k = \min_{c_i \subset CC_j} M(c_i)\} \quad (4)$$

The memory needed by the Smart Surface is $X \cdot M$, where X is the number of cells of the Smart Surface.

3.4 Execution time

Differentiating the parts is done in a distributed manner in several steps by each cell:

1: part reconstruction by each cell
2: **repeat**
3: criterion value computing (of the part on the Smart Surface)
4: comparison with all the models
5: **until** not differentiation
6: move part (taking of the same decision)

The aim of the framework is to find the best criteria. Hence, it is not involved in the image reconstruction and the move part; besides it is identical to all the criteria and depends on the distributed algorithm used, so it is not taken into account.

Suppose CC_i the best combination of criteria. The worst case is when *all* the criteria included in CC_i are needed to recognize the part among the models.

Let $t_1(c_j)$ be the execution time of the criterion c_j. Practical execution times of criteria is given in section 4.3. The total execution time to compute the value of all the criteria is: $T_1 = \sum_{c_j \subset CC_i} t_1(c_j)$.

The value of each criterion is a number. Comparing two numbers is very fast, let t_2 be the (constant) comparison time. n is the number of parts to differentiate. The execution time of all the comparisons is: $T_2 = \sum_{c_j \subset CC_i} nt_2$.

The total execution time is the sum of the two previous times:

$$\begin{aligned} T &= \textstyle\sum_{c_j \subset CC_i} (t_1(c_j) + nt_2) \\ &= nt_2 |CC_i| + \textstyle\sum_{c_j \subset CC_i} t_1(c_j) \end{aligned} \quad (5)$$

where $|CC_i|$ is the cardinality (number of elements) of CC_i.

4 Tests

The aim of our work is to differentiate relatively small parts by finding a set of criteria. These parts are represented by square matrices of order 3 or 4. All criteria are tested in order to find criteria or combination of criteria reaching total differentiation. These are called *total differentiation criteria*. Among these criteria, the fastest execution time and/or the lowest memory cost are selected.

4.1 Description of the criteria

The differentiation criteria must be simple and must be easy to implement. The criteria can be classified as contour-based methods or as region-based methods. For example, the first criterion, P (the perimeter) which is the number of cell frontiers between "1" and "0" (see fig. 7) is one of the simplest criteria, is classified as a contour-based method. The second criterion S is the area, classified as a region-based method. It consists in counting all the "1" contained in a part. In the following, the description and classification of the criteria used in our approach are given.

Contour-based criteria:

- P: The number of 1 having at least one neighbor at 0.
- A: The number of 1 having at least three neighbors to 0 and forming a right angle.

Region-based criteria:

- S: The number of 1 of the part.
- L: The maximum length between 1 of the part.
- N: The sum of the number of bits that change between two successive lines respectively columns.
- Z: The maximum length between all the 0 of part.
- D: The sum of 1 located on both diagonals.
- F: The sum of all Manhattan distances between 0.
- M: The sum of the number of bits that change.
- R: The sum of the number of V shape angles.
- I: The sum of the number of identical lines with the number of identical columns.
- T: The product of all Manhattan distances between 0.
- Y: The product of all Manhattan distances between 1.
- E: The product of the number of bits that change between each two successive lines with the number of bits that change between each two successive columns.
- K: The product of the number of bits that change from: the first line with the other lines, the last line with the other lines, the first column with the other columns, the last column with the other columns.
- C: The sum of the number of V shape angles.

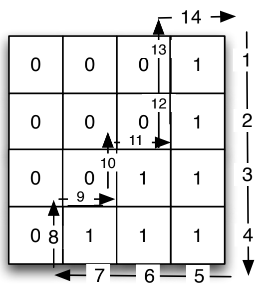

Figure 7. Perimeter of an object, equal to 14.

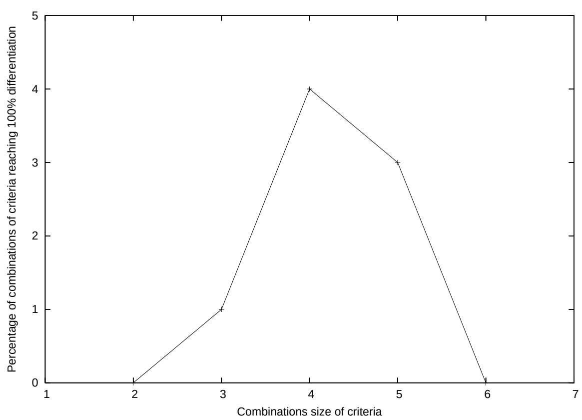

Figure 8. Size of combination reaching at 100%.

4.2 Selecting criteria reaching total differentiation

Among all the combinations of criteria, only the combinations reaching total differentiation are considered. The test show that the minimal combinations of criteria for matrices of size 3×3 are:
$CC_i = \{\{TM\}, \{TK\}, \{YF\}, \{YM\}, \{YK\}, \{YE\}\}$
and for matrices of size 4×4 are:
$CC_i = \{\{CFIDMRZ\}, \{CFILMRZ\}\}$.

Fig. 8 presents the number of combination of criteria reaching total differentiation function of the size of the combination. For 3×3 matrices all combinations of size 2 are removed from the combinations of size 3. For example (T,M) and (Y,E) reach 100%, therefore combinations ATM and AYE have been removed because they provide no additional differentiation. It's the same for all combinations.

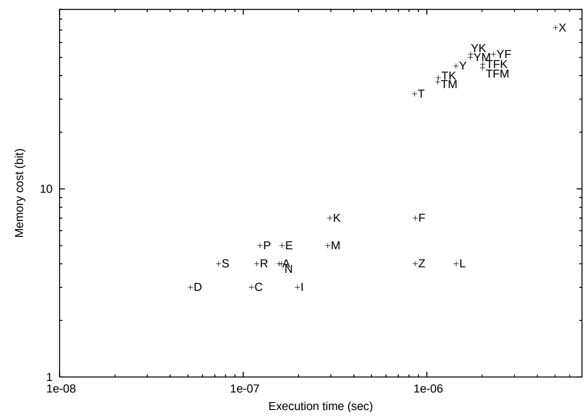

Figure 10. Memory cost according to execution times of criteria.

4.3 Memory costs and execution times of the criteria reaching 100%

To sum up, for 3×3 matrices six combinations of two criteria, among the criteria that we are defined, reach a total differentiation. However, it is obvious that the binary representation criterion, together with the grid based method [21], is sufficient to differentiate the parts. Although it is very costly in memory because the whole matrix is saved, i.e. 9 bits, 90° rotation matrices and mirrors matrices must also be saved. This gives 72 bits. Fig. 9 shows an example of memory consumption for all combination of the criteria T, Y, F, M, K, X.

Fig. 9 demonstrates that the combination TM reaches total differentiation with 37 bits, less than the binary representation. Execution times necessary for each criterion are measured. In fig. 10 the scatter of points of memory cost is presented function of execution time of criteria. There are several combinations of criteria that reach total differentiation with lower execution time and memory cost than binary representation.

5 Related work

Several methods of shape representation exist in the literature. They are divided into two categories: *contour-based* methods and *region-based* methods. *Contour-based* methods are widely used. But generally, for complex images, the contour is not enough to describe the image content, therefore *region-based* methods are better.

5.1 Contour-based approaches

In the contour-based approach, the pixels of the contour are considered.

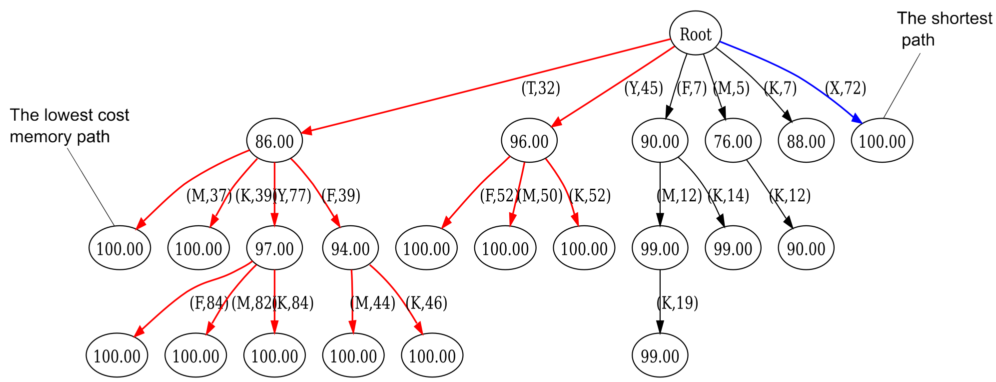

Figure 9. Memory cost.

5.1.1 Fourier descriptors

This approach is divided into two steps:

1. The image is defined by a one-dimensional function called *shape signature*, which is nothing else than a compact representation of the image [13]. Many methods to calculate the signature have been developed. The most common shape signatures are: centroid distance [24, 25], chordLength signature [26] and area function.

2. Once the shape signature has been calculated, a Fourier transform is applied [7, 5]. It results in coefficients called Fourier descriptors of the shape. These descriptors represent the shape of the object in the frequency domain. The Fourier transform is invariant against translation, scale, rotation and their starting point.

5.1.2 Freeman code

Freeman coding consists in browsing the borders of the shape with elementary moves from a starting point and coding the movement [6, 3].

Freeman code is sensitive to rotation because Freeman code depends on the starting point. To reduce this dependence, the resulting number has to be the minimal. The Freeman code is invariant to translation. It is also invariant to a rotation of 90° for the 4-connectivity and 45° for the 8-connectivity [8, 23].

Fourier descriptors and Freeman code are widely used for big pictures where the outline of the image differs noticeably from the inside of the images (parts). In our study these methods are not very interesting given that we are working on tiny images where the contour is equal or nearly equal to the surface.

5.2 Region-based approaches

In region-based methods, all the pixels within a shape are taken into account to obtain the shape representation.

5.2.1 Grid based

In this method [20], a fixed-length grid of cells on the image is draw. Going along our grid from top to bottom and from left to right, each cell that is wholly or partly covered by the form is affected with the value 1, and others cells with 0 [21]. This produces a binary number, which is the representation of our shape. The difference between two parts is given by an XOR between their binary representations. Such a binary representation is very sensitive to rotation, translation and dilatation, that is it requires a prestandardization.

5.2.2 Invariant moments

In this method [19, 16, 4], the invariant moments are used to represent the image. There are a set of seven descriptors called Husont invariants computed by normalizing central moments of order three. They are invariant to object scale, translation and orientation. They are used as input vectors for the classification method. There are several classification methods, among them neural networks are the most widely used because of their fault tolerance, their ability of classification and their generalizability. The invariant moments are widely used in three dimension models or large images that need to be compacted. It is not very useful to apply this method in our case because the images are very small.

6 Conclusions and future works

In this article we presented an exhaustive framework allowing to identify the criteria reaching a total differentiation among a set of criteria. Our tests on groups of 3 parts show that some combinations of two criteria for matrices of size 3×3 reached a total differentiation. We have considered the memory cost and execution time of the criteria and combinations of criteria that achieve a total differentiation. We have made a comparative study of these results with the execution time and cost memory of the grid based method. We have deduced that some combinations of criteria reach a total differentiation with a smaller execution time and a lower cost memory than the grid based method.

One of the idea of our future work is to reduce the constraint of the $90°$ rotation by allowing a more flexible rotation (for instance a step-by-step rotation with a $10°$ step). Another idea is to develop a distributed algorithm for various criteria in order to implement them in the Smart Surface and compare them in term of execution time. Finally, we plan to implement the ECO framework on a G80 GPU with CUDA, in order to speed up the comparison.

References

[1] D. Biegelsen, A. Berlin, P. Cheung, M. Fromherts, D. Goldberg, W. Jackson, B. Preas, J. Reich, and L. Swartz. Airjet paper mover. In *SPIE Int. Symposium on Micromachining and Microfabrication*, 2000.

[2] K.-F. Böhringer, V. Bhatt, B. R. Donald, and K. Y. Goldberg. Algorithms for sensorless manipulation using a vibrating surface. *Algorithmica*, 26(3-4):389–429, 2000.

[3] S. Bres, J.-M. Jolion, and F. Lebourgeois. *Traitement et analyse des images numériques*. Hermés, 2003.

[4] C. Chen. Improved moment invariants for shape discrimination. *Pattern Recognition*, 26(5):683–686, May 1993.

[5] S. Derrode, M. Daoudi, and F. Ghorbel. Invariant content-based image retrieval using a complete set of Fourier-Mellin descriptors. In *ICMCS, Vol. 2*, pages 877–881, 1999.

[6] L. Diane. *Introduction au Traitement d'Images*. Vuibert, Paris, France, 2nd edition, Feb. 2008.

[7] R. Diaz de Léon and L. Sucar. Human silhouette recognition with Fourier descriptors. In *ICPR00, vol III*, pages 709–712, 2000.

[8] P.-E. Forssén and A. Moe. Contour descriptors for view-based object recognition. Technical Report LiTH-ISY-R-2706, Dept. EE, Linköping University, SE-581 83 Linköping, Sweden, sep 2005.

[9] H. Fujita. Group work of microactuators. In *International Advanced Robot Program Workshop on Micromachine Technologies and Systems*, pages 24–31, Tokyo, Japan, Oct 1993.

[10] Y. Fukuta, Y.-A. Chapuis, Y. Mita, and H. Fujita. Design, fabrication and control of mems-based actuator arrays for air-flow distributed micromanipulation. *IEEE Journal of Micro-Electro-Mechanical Systems*, 15(4):912–926, Aug 2006.

[11] K. Y. Goldberg. Orienting polygonal parts without sensors. *Algorithmica*, 10(2-4):210–225, 1993.

[12] C. Liu, T. Tsao, P. Will, Y. Tai, and W. Liu. A micromachined permalloy magnetic actuator array for micro robotics assembly systems. In *The 8th International Conference on Solid-State Sensors and Actuators*, 1995.

[13] S. Loncaric. A survey of shape analysis techniques. *Pattern Recognition*, 31(8):983–1001, 1998.

[14] J. E. Luntz and W. Messner. A distributed control system for flexible materials handling. *IEEE Control Systems*, 17(1), Feb 1997.

[15] J. E. Luntz, W. Messner, and H. Choset. Parcel manipulation and dynamics with a distributed actuator array: The virtual vehicle. In *IEEE Int. Conf. on Robotics and Automation (ICRA)*, pages 1541–1546, 1997.

[16] M. Mercimek, K. Gulez, and T. Velimumcu. Real object recognition using moment invariants. *Sadhana*, 30(6):765–775, 2005.

[17] G. C. Murilo and M. W. Peter. A general theory for positioning and orienting 2d polygonal or curved parts using intelligent motion surfaces. In *ICRA*, pages 856–862, 1998.

[18] K. Pister, R. Fearing, and R. Howe. A planar air levitated electrostatic actuator system. In *IEEE Workshop on Micro Electro Mechanical Systems*, pages 61–71, 1990.

[19] R. Prokop and A. Reeves. A survey of moment-based techniques for unoccluded object representation and recognition. *GMIP*, 54(5):438–460, sep 1992.

[20] A. Sajjanhar and G. Lu. A grid-based shape indexing and retrieval method. *Australian Computer Journal*, 29(4):131–140, 1997.

[21] C. Shahabi and M. Safar. An experimental study of alternative shape-based image retrieval techniques. *Multimedia Tools Appl.*, 32(1):29–48, 2007.

[22] J. Suh, S. Glander, R. Darling, C. Storment, and G. Kovacs. Combined organic thermal and electrostatic omnidirectional ciliary microactuator array for object positioning and inspection. In *Solid State Sensor and Actuator Workshop*, 1996.

[23] J. Sun and X. Wu. Shape retrieval based on the relativity of chain codes. In *MCAM07*, pages 76–84, 2007.

[24] D. Zhang and G. Lu. Content-based shape retrieval using different shape descriptors: A comparative study. In *ICME*. IEEE Computer Society, 2001.

[25] D. Zhang and G. Lu. Shape-based image retrieval using generic Fourier descriptor. *SPIC*, 17(10):825–848, Nov 2002.

[26] D. Zhang and G. Lu. Study and evaluation of different Fourier methods for image retrieval. *Image Vision Comput.*, 23(1):33–49, 2005.