Multidimensional Quick Invariant Signature Extraction

Hamid Reza Shahbazkia[a], António dos Anjos[a,b], Katarzyna Gębal[b]

[a] *University of Algarve*
Department of Electronics and Informatics Engineering
8005-139 Faro, Portugal
hshah@ualg.pt
[b] *DTU Informatics*
2800 Kgs. Lyngby, Denmark
aanjos@imm.dtu.dk, katg@imm.dtu.dk

Abstract

Simple geometric properties can be used to define the signature of a pixel pattern independently of its position, size or orientation in an image. These properties can be extended to obtain invariances in volumes or even in any N-dimensional data space. In this paper we present a simple and fast approach to create a Quick Invariant Signature (QIS), that can be used for usual affine transformations as well as for other transformations such as toric or cylindrical translation or window size modifications. The signature extraction is performed in data space and directly on points without any estimation or compression. We provide a formal demonstration that the signature is totally invariant in any N-dimensional space, studying the case of 2-D images, before extending the notion to many dimensional spaces.

Keywords: Shape matching, invariant descriptor, image signature.

1. Introduction

Data analysis often requires considering the possible transformations the data has been subjected to. In a vision system for example, invariance to translation is a usual and straightforward requirement. Many different approaches to obtain invariance in 2-D images for pattern recognition have been developed and largely used [8, 12, 4, 6, 7]. We can classify them into three main classes: Invariant moments, invariant descriptors, and invariant neural architectures. Invariant moments [1] and Fourier-Mellin descriptors [3, 10] require complex calculations and significant computation time. They are quite sensitive to noise and images cannot be directly analyzed. Therefore only a few of the interest points are taken into consideration. Other invariant descriptors, such as the circular descriptor, are highly sensitive to noise and can cause sampling problems. The most popular neural-network approach [11, 5], a multi-layer perceptron network and a back-propagation learning algorithm, requires a huge training set to learn the invariance to affine 2-D transformations. Other neural networks try to integrate invariance into their architectures but this causes a problem of network size by combinatorial explosion that seriously limits the size of their input. The Quick Invariant Signature (QIS) extraction proposed here is a formal method which uses simple geometric notions such as length and slope to define the invariant signature of a binary image. The size of the input is not limited and no pre-processing is needed. The only requirement is that the image is binary, for example, the result of an edge detection operation. The whole image is taken into account for the extraction of the signature. In [9] we presented an approach to define a binary image in terms of active couples of points as well as algorithms to extract the signatures. The multidimensional QIS presented here simplifies and extends these approaches. QIS can be used to extract a signature, not only invariant to classic affine 2-D transformations such as translation, rotation and scale, but also to unusual transformations such as symmetry, cylindrical, toric translations and change of fovea size. QIS can also be applied directly to rectangular images. We demonstrate that QIS is totally invariant to the considered transformations. QIS can even be integrated into a neural network architecture to provide 100% invariant neural networks or be used to test, in a formal approach, if two classes of forms are "invariant separable".

2. Invariances

When an image or pattern is transformed, some evident low-level properties such as lengths, slopes and angles between couples or triplets of points remain unchanged. For example, it is easy to demonstrate that the lengths and slopes of couples of points remain unchanged after translation. Table 1 lists some invariants.

Instead of defining a binary image by fore- and background pixels, we can define it in terms of couples of foreground pixels, and we can use the

Transformation	**Invariant**
Translation	Distances + Slopes
Translation + Rotation	Distances
Translation + Scale	Slopes
Translation + Rotation + Scale	Triangles

Table 1: Transformations *vs.* invariants.

invariants of Table 1 to extract the invariant signature. In the next sections, foreground pixels and couples of pixels are called *active* pixels or couples.

3. Signature Extraction

When an image is defined in terms of similar couples of points, we can obtain an invariant signature. Similar couples are those which present the same mathematical properties defining an invariance. For example, invariance to translation is defined by identical length and slope. In this case, similar couples are all couples with the same length and the same slope. The signature is then obtained by counting the number of similar couples in each similarity group. The details of this process are given case by case in the following sections.

3.1. Translation QIS

As explained above, in the case of translation invariance, the similarity properties taken into account are the lengths and slopes of the active couples of points. The first step is to count the similarity groups. We can demonstrate that on an $n \times n$ grid, the number of different length-slope combinations is $2n(n-1)$ (see Fig. 1).

There are $(n-1)^2$ different combinations with a negative slope, $(n-1)^2$ with a positive slope, $n-1$ horizontal, and $n-1$ vertical. The invariant signature of the image is obtained by counting the number of active couples in each of these numbered groups. The result is a table of $2n(n-1)$ integers where the index is the order of the similarity group and the value is the number of active couples in this group. Hence, the size of the signature is always less than twice the size of the image.

Two examples are given in Figs. 2 and 3. In Fig. 2, only one couple of points is active on the 4×4 grid. There are 24 different similarity groups and the order of the group of the active couple is 5. Then the signature of this image will be:

$$\begin{array}{ccccccc}
 & & & + & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0
\end{array}$$

All the translated views of this couple will give the same signature since the order of the similarity group of the active couple remains 5. Fig. 3 shows a 9×9 grid with 144 similarity groups. All the active couples are counted to define the invariant signature. The following array shows the translation invariant signature of Fig. 3:

$$\begin{array}{ccccccccccccccccc}
 & & & & & & & & + & 9 & 6 & 3 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 2 & 2 & 2 & 7 & 2 & 2 & 2 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 2 & 2 & 2 & 5 & 2 & 2 & 3 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 2 & 3 & 4 & 6 & 4 & 3 & 4 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 2 & 3 & 5 & 4 & 3 & 4 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 2 & 2 & 3 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 2 & 2 & 2 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 2 & 3 & 4 & 3 & 2 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{array}$$

In fact this array can be considered as a redefinition of an image in a new 3-D space having as axis the length, the slope and the number of couples. This 3-D space can be reduced to a 2-D space where the area covered by the slope and the length is represented by the similarity group order axis. To find to which group a couple of points belongs, one should compare the length

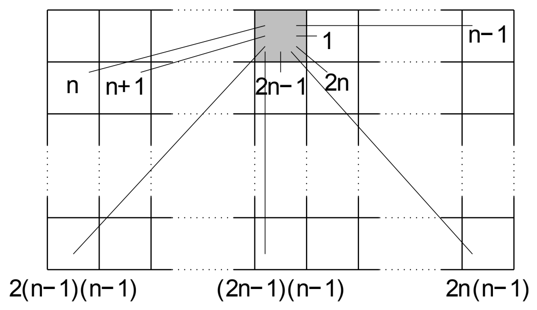

Figure 1: Possible slope combinations.

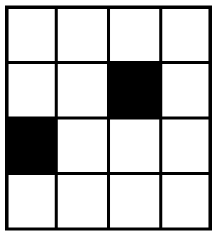

Figure 2: One active couple on a 4×4 grid.

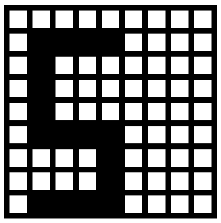

Figure 3: Multiple active couples on a 9×9 grid.

and slope of this couple to the length and slope of each similarity group. Considering an $n \times n$ grid, we will have at most $2n(n-1)-1$ comparisons per couple and at most $(n^2!)/(2!(n^2-2)!)$ couples. Therefore, this comparison process for extracting the signature is rather slow. For this reason we have formalized the access to the order number of the similarity groups by a simple function that eliminates the sequential comparison process.

Given two pixels p and q where the subscripts r and c refer to the row and the column, respectively, Algorithm 3.1 will find the similarity group number (SGN) s of a couple.

Algorithm 3.1: GET-SGN-T(p, q, n)

```
$s \leftarrow |q_c - p_c + (2n-1)(q_r - p_r)|$
**return** $(s)$
```

This function yields the same number for all couples with equal slopes and equal lengths. Signature extraction is then performed by Algorithm 3.2, where $\mathcal{A}$ is the set of all active couples (p, q).

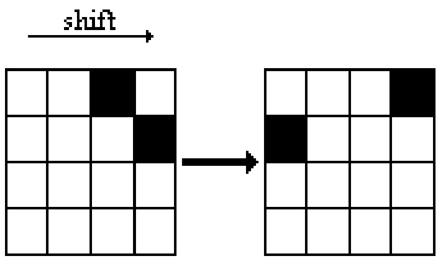

Figure 4: Cylindrical translation.

Algorithm 3.2: EXTRACTSIGNATURE-T$(\mathcal{A}, n)$

```
local $t[2n(n-1)]$
for $i \leftarrow 0$ to $2n(n-1)$
  do $t[i] \leftarrow 0$
for each $(p, q) \in \mathcal{A}$
  do { local $s \leftarrow$ GET-SGN-T$(p, q, n)$
     { $t[s] \leftarrow t[s] + 1$
return $(t)$
```

From here on, we can use the translation-invariant signature to define many other invariances, such as cylindrical or toric translation.

3.2. Cylindrical Translation QIS

Cylindrical translation (CT) invariance may seem useless in pattern recognition, but one of its possible applications will be shown below. Although the formal definition of a CT is quite difficult to establish, its expression in terms of similarity groups is rather simple. Fig. 4 shows two possible positions of a couple of points in a CT. It can be seen that one couple of points in a CT can belong, at most, to two similarity groups of a simple translation. In this case the length of the couple and its slope are not unchanged by the CT anymore. If the two possible simple translation similarity groups are considered as the same similarity group for the CT, we can easily express the invariance to CT. QIS extraction remains exactly the same except for the SGN Algorithm. In this case, the additional SGN-CT function should find the second similarity group order number on the basis of the first number. When the couple is vertical, *i.e.* , perpendicular to the translation direction, it will only define one similarity group. Therefore, QIS extraction for a CT uses the same similarity groups as the simple translation, but the counting of combinations belonging to each similarity group is different.

After applying Algorithm 3.1, Algorithm 3.3 yields the correct order number of these groups.

Algorithm 3.3: GET-SGN-CT(s, n)

```
if $s < n$
  then $t \leftarrow n - s$
  else $t \leftarrow s - n[\mathbf{signum}\{(s-n)\mathbf{mod}(2n-1) - (n-1)\}]$
return $(t)$
```

Signature extraction may be performed by Algorithm 3.4, whereas the size of the signature will still be $2n(n-1)$.

Algorithm 3.4: EXTRACTSIGNATURE-CT$(\mathcal{A}, n)$

```
local $t[2][2n(n-1)]$
for $i \leftarrow 0$ to $2$
  do { for $j \leftarrow 0$ to $2n(n-1)$
         do $t[i][j] \leftarrow 0$
for each $(p, q) \in \mathcal{A}$
  do { local $s[1] \leftarrow$ GET-SGN-T$(p, q, n)$
       local $s[2] \leftarrow$ GET-SGN-CT$(s[1], n)$
       $t[1][s[1]] \leftarrow t[1][s[1]] + 1$
       $t[2][s[2]] \leftarrow t[2][s[2]] + 1$
return $(t)$
```

3.3. Toric Translation QIS

Toric translation (TT) invariance can be used for regular texture recognition. As for the CT case, we will use the simple translation similarity groups to define the TT signature. The only thing to be added is the SGN-TT function and the number of similarity groups involved in a TT. Fig. 5 shows four possible positions of a couple of points after a TT. Each couple of points in a TT can belong, at most, to four similarity groups of a simple translation. In this case, QIS extraction follows the same reasoning as for CT. The SGN function for TT QIS extraction is presented in algorithm 3.5.

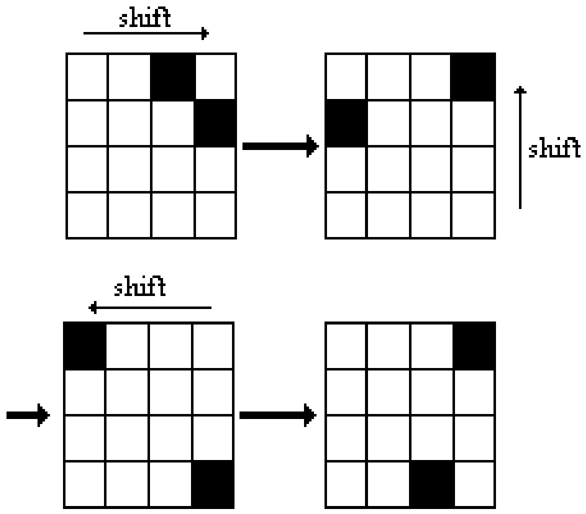

Figure 5: Toric translation.

Algorithm 3.5: GET-SGN-TT(s, n)

```
if $s < n$
  then $t \leftarrow s$
  else $t \leftarrow (2n-1)n - s$
return $(t)$
```

This algorithm together with the previously described ones are used to determine the numbers corresponding to the four possible configurations. As a result, we can find the same numbers for all couples belonging to each of the four similarity groups. Signature extraction is then done as described by algorithm 3.6. The size of the signature still remains $2n(n-1)$.

Algorithm 3.6: EXTRACTSIGNATURE-TT($\mathcal{A}, n$)

```
local $t[4][2n(n-1)]$
for $i \leftarrow 0$ to $4$
  do { for $j \leftarrow 0$ to $2n(n-1)$
         do $t[i][j] \leftarrow 0$
for each $(p, q) \in \mathcal{A}$
  do { local $s[1] \leftarrow$ GET-SGN-T$(p, q, n)$
       local $s[2] \leftarrow$ GET-SGN-CT$(s[1], n)$
       local $s[3] \leftarrow$ GET-SGN-TT$(s[1], n)$
       local $s[4] \leftarrow$ GET-SGN-TT$(s[2], n)$
       $t[1][s[1]] \leftarrow t[1][s[1]] + 1$
       $t[2][s[2]] \leftarrow t[2][s[2]] + 1$
       $t[3][s[3]] \leftarrow t[3][s[3]] + 1$
       $t[4][s[4]] \leftarrow t[4][s[4]] + 1$
return $(t)$
```

3.4. Symmetry QIS

Symmetry invariance can also be coded by using the simple translation similarity groups and a new SGN function. Fig. 6 shows two possible symmetric positions of a couple of points. One couple of points in a symmetry can belong, at most, to only two similarity groups of a simple translation. Algorithm 3.7 computes the two similarity groups involved in symmetry invariance.

Algorithm 3.7: GET-SGN-S(s, n)

```
if $s < n$
  then $t \leftarrow s$
  else $t \leftarrow s - 2\{(s-n)\mathbf{mod}(2n-1) - (n-1)\}$
return $(t)$
```

3.5. Size Invariant QIS

In some cases, the size of the grid containing the reference pattern is not equal to the size of the grid containing the test pattern. QIS extraction can easily be adjusted to obtain invariance to grid size. QIS of Fig. 7(a) may be used to classify that of Fig. 7(b), or vice-versa. The idea is to extract the

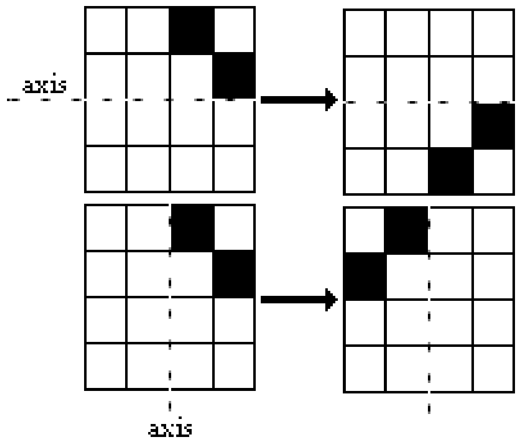

Figure 6: Symmetry QIS.

QIS of the test image by using the size of the reference image. Let $n \times n$ be the size of the test image and $m \times m$ be that of the reference image.

If s is the order number of the similarity group of a couple on the $n \times n$ grid, then s is found by one of the SGN functions defined earlier. The next step consists in finding the order number of the same group on the $m \times m$ grid. This is done by the following function:

Algorithm 3.8: GET-SGN-FS(s, m, n)

```
$t \leftarrow s + 2\lfloor \frac{s+n-1}{2n-1} \rfloor \rfloor (m - n)$
**return** $(t)$
```

QIS extraction is then performed by algorithm 3.9, where EXTRACTSIGNATURE can be any of the signature extraction functions defined earlier.

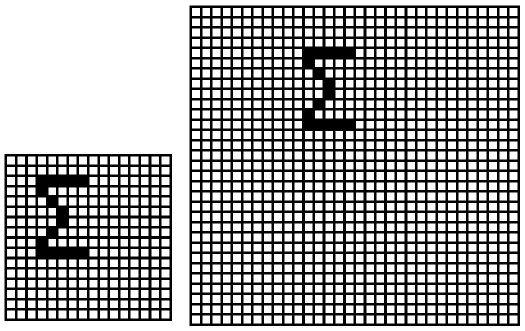
(a) 16×16. (b) 32×32.

Figure 7: Different size grids with same pattern.

Algorithm 3.9: EXTRACTSIGNATURE-FS($\mathcal{A}, m, n$)

```
**local** $t[2m(m-1)]$
**local** $r[2n(n-1)]$
$r \leftarrow$ EXTRACTSIGNATURE($\mathcal{A}, n$)
**for** $i \leftarrow 0$ **to** $2m(m-1)$
  **do** $t \leftarrow 0$
**for** $i \leftarrow 0$ **to** $2n(n-1)$
  **do** **local** $s \leftarrow$ GET-SGN-FS($i, m, n$)
     **if** $s < 2m(m-1)$
       **then** $t[s] \leftarrow r[i]$
**return** ($t$)
```

3.6. Translation, Rotation and Scale QIS

Here we choose similar triangles as invariant primitives in the case of translation, rotation and scale (TRS) invariance. QIS extraction cannot be directly performed because counting the number of different triangles in an $n \times n$ grid is impossible. This problem seems to be open and, to the best of our knowledge, no formal solution exists. Therefore we apply a simple and well-known log-polar space conversion to achieve TRS invariance. This conversion modifies a rotation to a cylindrical translation.

Let $P(\rho, \theta)$ be the polar coordinates of a point $p(x, y)$. In the polar imaginary space we will have $Z = \rho e^{\imath\theta}$. If P is rotated by α then $P_r(\rho, \theta + \alpha)$ and $Z_r = \rho e^{\imath(\theta+\alpha)}$. If P is scaled by a factor k then $P_s(k\rho, \theta)$ and $Z_s = k\rho e^{\imath\theta}$.

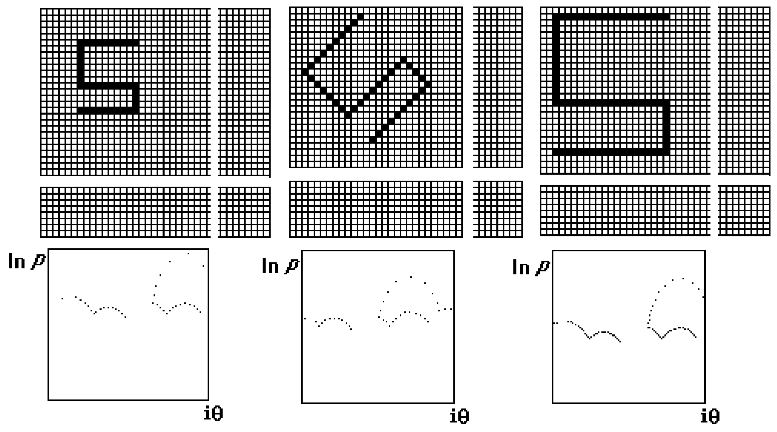

Figure 8: log-polar transformation.

Now let us consider the logarithm of Z:
$Z = \rho e^{\imath\theta} \implies \ln Z = \ln\rho + \imath\theta$.
Then
$Z_r = \rho e^{\imath(\theta+\alpha)} \implies \ln Z_r = \ln\rho + \imath(\theta + \alpha)$ (translation along the $\imath$ axis);
$Z_s = k\rho e^{i\theta} \implies \ln Z_s = \ln k + \ln\rho + \imath\theta$ (translation along the $\ln\rho$ axis).
A translation in $(\ln\rho, \imath\theta)$ space defines a rotation and/or a scale change in (x, y) space. The only problem with this method is that the translation along the $\imath\theta$ axis is periodical, which implies that the translation along the $\imath\theta$ axis in the $(\ln\rho, \imath\theta)$ space is $\mod 360°$. Our QIS extraction for cylindrical translation is especially useful here. The procedure to obtain the QIS for TRS is as follows:

1. Perform the space conversion to obtain the *log*-polar signature LPS;
2. Extract the signature from the LPS matrix using Algorithm 3.4.

Fig. 8 shows three examples of different images transformed into log-polar coordinates.

3.7. Non-square grids

So far we have discussed the case of square grids of $n \times n$. As to non-square grids, one quick solution would be to append zeros to the smallest dimension, which makes the matrix square and, then apply the already presented methods. However, we have developed a method that allows for direct processing of such grids. If we apply the space transformation presented in

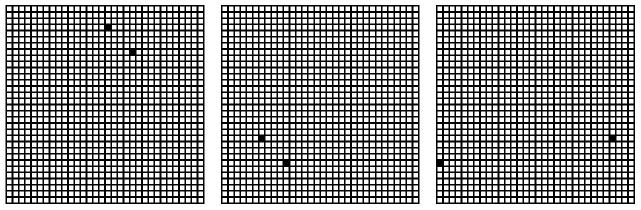

Figure 9: One active couple in each image.

Section 3.6, all we need is to do is redefine the translation invariant method for non-square grids, thus allowing the same reasoning applied to square grids to be applied to non-square ones. As the number of similarity groups in an $n \times m$ grid is $(n \times m - 1) + (n - 1)(m - 1)$, most of the functions presented before do not need to be changed because they depend on the width of the image which is n. The only exception is the toric translation SGN which requires both the width n and the height m of the grid.

Algorithm 3.10: GET-SGN-TT-NS(s, n, m)

```
if $s < n$
  then $t \leftarrow s$
  else $t \leftarrow (2n-1)m - s$
return $(t)$
```

The direct manipulation of the non-square matrices allows to set a different partitioning for each of the two axis (angle, logarithm) and get the best results according to the considered transformation.

Table 2 presents three examples of execution of the several discussed SGN functions to the transformations of Figs. 9(a), 9(b) and 9(c).

4. Adding more dimensions

The QIS computation can be extended to more dimensions where most of the algorithms developed for the two-dimensional case can still be applied. Adding a new dimension is performed by applying two-dimensional SGN computations to the space that has one of the coordinates corresponding to an order of a group of the less-dimensional problem and the second coordinate corresponding to the added dimension.

Figure	**9(a)**		**9(b)**		**9(c)**	
Pairs	**p**	**q**	**p**	**q**	**p**	**q**
r	4	8	22	26	1	29
c	17	21	7	11	26	22
i	113	245	679	811	801	701
T	132		132		1864	
CT	[132, 1864]		[132, 1864]		[1864, 132]	
TT	[132, 1864, 1144, 924]		[132, 1864, 1144, 924]		[1864, 132, 1144, 924]	
S	[132, 1120]		[132, 1120]		[1864, 156]	

Table 2: SGN of the transformations.

We note here that it is possible to have different pixels with the same indexes in a lower-dimensional space. This problem is solved by allowing the signature group number to be zero in the lower-dimensional space. As a result, the number of similarity groups for an $m \times n \times o$ grid is equal to the number of similarity groups for the $(p+1) \times o$ grid, where p is the size of the $m \times n$ group.

Another issue is that, for a pair of points, the ordering of them should be traced. This is done by preserving the sign of the similarity group number returned by the basic two-dimensional function. One can, at the end, get rid of this sign by taking the absolute value in the final dimension. Figs. 10 and 11 illustrate how the calculation of a similarity group is done in two particular situations. The modified two-dimensional SGN function is presented below:

Algorithm 4.1: GET-SGN-2D(p, q, n)

```
$s \leftarrow q_c - p_c + (2n-1)(q_r - p_r)$
**return** $(s)$
```

The N-dimensional version of the function takes three vectors as parameters: p and q which contain the points' coordinates in all dimensions and n which specifies the size of the grid in each dimension.

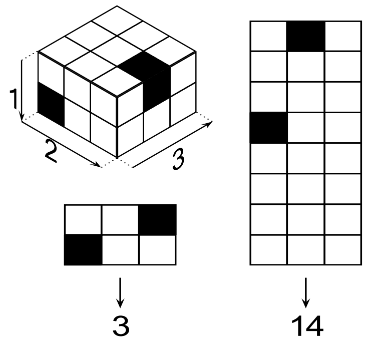

Figure 10: Calculation of a similarity group number for an active couple in $2 \times 3 \times 3$ grid. First a similarity group number is extracted from 2×3 subspace. This signature is then used to position the pixels on an 8×3 grid which is used to calculate the final SGN.

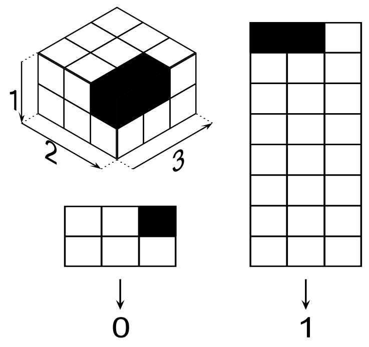

Figure 11: Calculation of a similarity group number of an active couple on a $2 \times 3 \times 3$ grid in the case when the pixels coincide in the first two coordinates.

Algorithm 4.2: GET-SGN-MD(k, p, q, n)

```
local $p', q'$
local $s' \leftarrow 1$
if $k > 2$
  then
    $s' \leftarrow$ GET-SGN-MD$(k-1, p, q, n)$
    $p'_r \leftarrow 1$
    $q'_r \leftarrow 1 + |s'|$
    if $s' < 0$
      then
        $q'_c \leftarrow p[k]$
        $p'_c \leftarrow q[k]$
      else
        $q'_c \leftarrow q[k]$
        $p'_c \leftarrow p[k]$
```

```
  else
    $p'_r \leftarrow p[1]$
    $p'_c \leftarrow p[2]$
    $q'_r \leftarrow q[1]$
    $q'_c \leftarrow q[2]$
$s \leftarrow$ signum$(s')$GET-SGN-2D$(p', q', n[k])$
return $(s)$
```

4.1. Recalculating the SGN in many dimensions

Transformations such as shift and symmetry in a many-dimensional SGN require the specification of the coordinate in which the transformation occurs. In order to re-compute the order number of a group after such a transformation, when the original SGN is given, one needs to "peel" off the higher dimensions and obtain the order number of the group in the required dimension "peeled off" dimensions. Then signature extraction is applied using two-dimensional algorithms. Finally, the higher dimensionality is restored.

In signature computation we take the pair ordering into account, as this may change due to the shift or symmetry transformations. This information is stored in the sign of the number returned by the recursive function. The final SGN will be obtained by taking the absolute value.

Below is the recursive algorithm which calculates the SGN after a shift operation in coordinate i. Here, k is the number of dimensions of the space for which a signature s was computed and n is the vector holding the size of each dimension.

Algorithm 4.3: GET-SGN-C-MD(i, s, k, n)

```
if $i = k$
  then { $t \leftarrow$ GET-SGN-CT$(s, n[k])$
         if $t < n[k]$ and $t > 0$
           then $t \leftarrow -t$
  else { local $s' \leftarrow \lfloor \frac{s+n[k]-1}{2n[k]-1} \rfloor$
         local $t' \leftarrow$ GET-SGN-C-MD$(i, s', k-1, n)$
         $t \leftarrow s + (|t'| - s')(2n[k] - 1)$
         if $t' < 0$
           then $t \leftarrow -$GET-SGN-S$((t, n[k]))$
return $(t)$
```

We note that the algorithm also works for for a shift in coordinate $i = 1$ (see Fig. 13), which, in the two-dimensional case, corresponds to function GET-SGN-TT.

A similar solution can be applied if we want to get the SGN in case of a symmetry transformation. Compared to the previous example, the only thing that changes is the function for recalculating the SGN for a 2-D transformation, which is symmetry in this case.

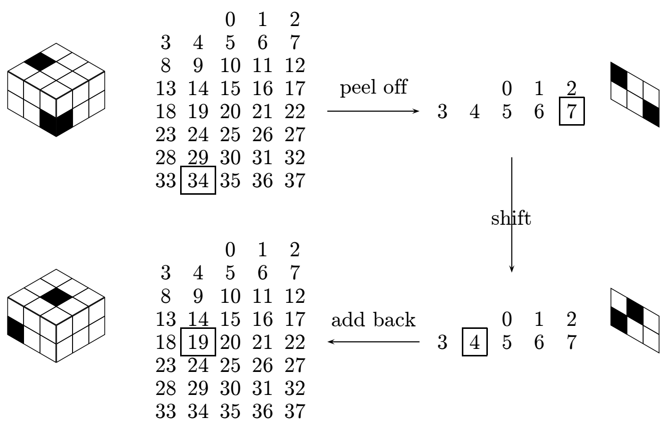

Figure 12: Recalculation of the SGN for a shift operation in the coordinate $i = 2$. All possible similarity group numbers are shown in the matrices. The active number is indicated by a frame. The "peel off' operation can be seen as taking the row number (starting from 0) of the signature position in the table. The "add back" is then performed by changing the row according to the lower dimensional signature, but keeping the column of the previous signature.

Algorithm 4.4: GET-SGN-S-MD(i, s, k, n)

```
if $i = k$
  then { $t \leftarrow$ GET-SGN-S$(s, n[k])$
         if $t < n[k]$ and $t > 0$
           then $t \leftarrow -t$

  else { local $s' \leftarrow \lfloor \frac{s+n[k]-1}{2n[k]-1} \rfloor$
         local $t' \leftarrow$ GET-SGN-S-MD$(i, s', k-1, n)$
         $t \leftarrow s + (|t'| - s')(2n[k] - 1)$
         if $t' < 0$
           then $t \leftarrow -$GET-SGN-S$((t, n[k]))$
return $(t)$
```

In a similar way, the grid size can be changed. We only need to provide an additional parameter which is the modified grid size m for a given coordinate number i. This operation does not change the pixel ordering. Therefore, the pseudo-code that deals with the orientation consistency can be skipped.

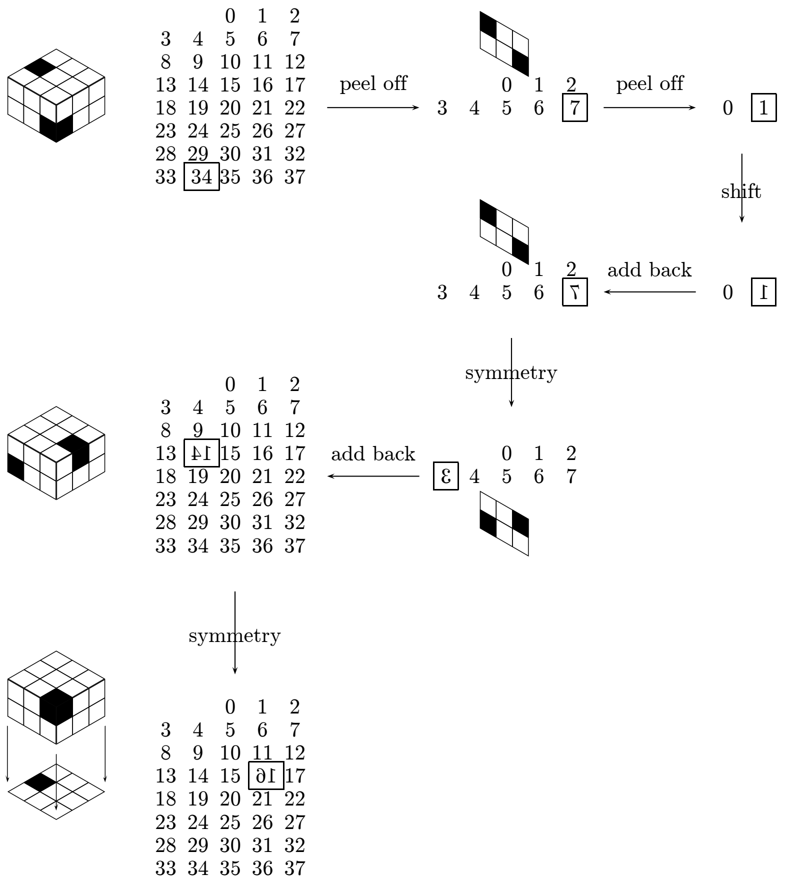

Figure 13: Recalculation of the SGN for a shift operation in the coordinate $i = 1$. Note that, after a shift recalculation, the SGN is the same but the orientation has changed, which is indicated by reversing the content of the frames. Consequently, an additional symmetry operation is performed in upper spaces.

Algorithm 4.5: GET-SGN-FS-MD(i, m, s, k, n)

```
if $i = k$
  then $t \leftarrow$ GET-SGN-FS$(s, m, n[k])$
  else $\begin{cases} \textbf{local } s' \leftarrow \lfloor \frac{s+n[k]-1}{2n[k]-1} \rfloor \\ \textbf{local } t' \leftarrow \text{GET-SGN-FS-MD}(i, m, s', k-1, n) \\ t \leftarrow s + (|t'| - s')(2n[k]-1) \end{cases}$
return $(t)$
```

The functions presented above may be considered as basic building blocks. By combining them, one can obtain a method to recompute a group's order number for more complicated operations. For example, if we want to get the SGN after shifts in both coordinates i_1 and i_2, then the result is obtained by: GET-SGN-C-MD $(i_2$,|GET-SGN-C-MD $(i_1$,s,k,$n)$|,k,$n)$.
This yields a correct SGN but may be computationally inefficient. Below we present an algorithm that recalculates the similarity group number for simultaneous shift operations in many dimensions. Here i is a boolean vector and for a given coordinate k, $i[k]$ indicates wether the shift in coordinate k needs to be performed. Similar versions can be made for symmetry and grid-size change.

Algorithm 4.6: GET-SGN-C-MD-V(i, s, k, n)

```
local $t \leftarrow 1$
if $k > 1$
  then $\begin{cases} \textbf{local } s' \leftarrow \lfloor \frac{s+n[k]-1}{2n[k]-1} \rfloor \\ t' \leftarrow \text{GET-SGN-C-MD-V}(i, s', k-1, n) \\ s \leftarrow s + (|t'| - s')(2n[k]-1) \end{cases}$
if $i[k]$
  then $\begin{cases} s \leftarrow \text{GET-SGN-CT}(s, n[k]) \\ \textbf{if } s < n[k] \textbf{ and } s > 0 \\ \quad \textbf{then } t' \leftarrow -t' \end{cases}$
if $t' < 0$
  then $s \leftarrow -$GET-SGN-S$((s, n[k]))$
return $(t)$
```

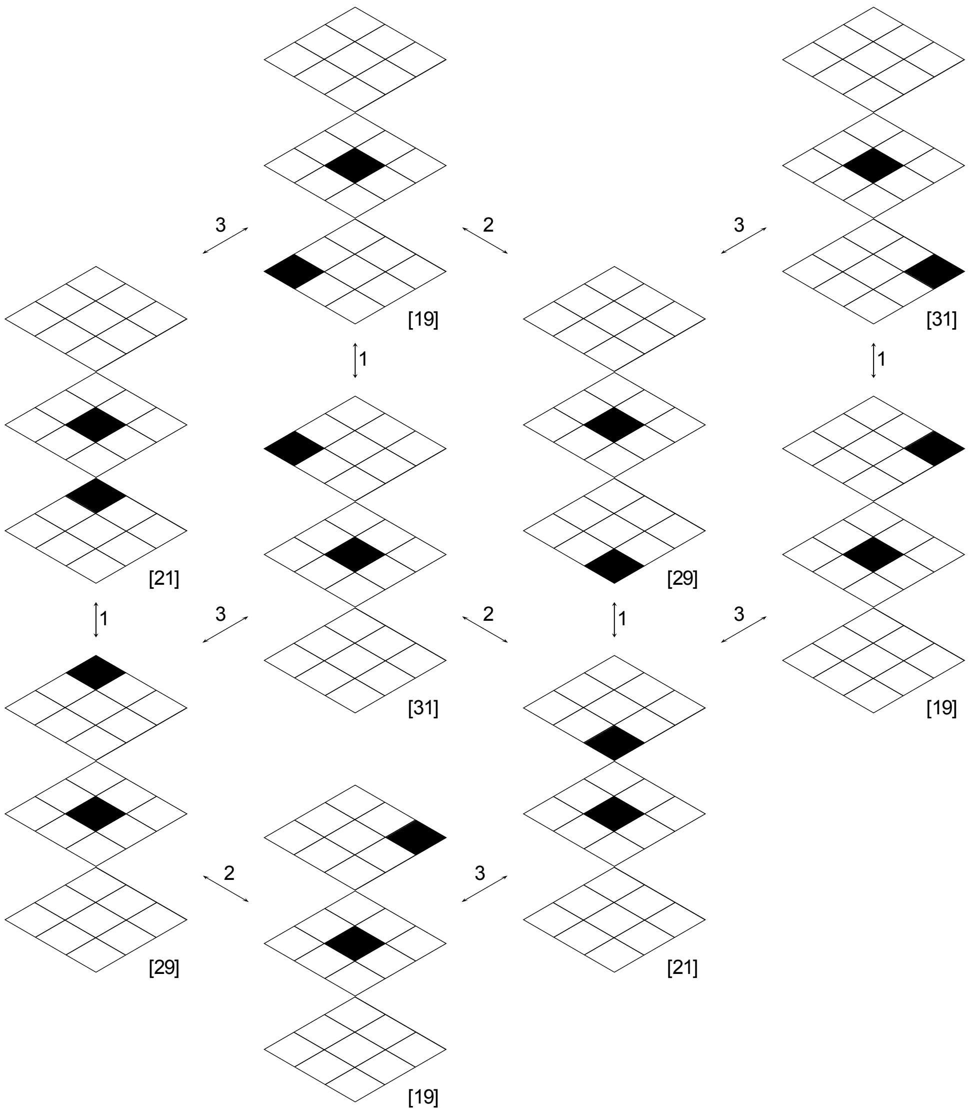

Figure 14: Symmetry operations performed on one active couple. In the square braces note repeating similarity group numbers even if the positions of active pixels are different. After performing symmetry in all three dimensions the result has the SGN of an original image. In general, for a n-dimensional problem, one can have maximally $2^k - 1$ unique similarity group numbers obtained by performing any combination of symmetry on an initial active couple. This number is equal to the number of diagonals of n-dimensional cube. For a shift operation one can obtain 2^k SGN numbers.

5. Conclusions

As demonstrated above, QIS is a very robust and time-saving method to extract a signature from a binary image and it also includes an extension to many dimensions. Nevertheless, it is difficult to formally demonstrate the uniqueness of QIS: a few totally symmetric patterns which produce the same QIS can be found. Of course, the uniqueness of QIS in case of asymmetrical patterns is intuitively evident. Even if the uniqueness of QIS can never be formally demonstrated, in practice the probability of finding the same QIS for two different patterns is nearly zero.

Another aspect to consider is related to the presence of noise. As QIS extraction uses couples of points, the introduction of one "parasite" pixel in an image, with a total of n pixels, will introduce n parasite couples. As the image is defined in terms of couples there will be a maximum of $n!/2!(n-1)!$ couples. Therefore, when the noise of an image is of the order of $1/n$ pixels, the number of couples will be $2/(n-1)$. It can be considered, then, that the perturbation has as factor of two. In spite of this theoretical disadvantage, QIS remains in practice much more robust in the presence of noise than other invariant recognition methods such as invariant moments [2]. This is because QIS preserves the signature of the original image on a noisy grid. If we consider an image of n pixels and one parasite pixel, the QIS extraction will show the QIS of the image incremented by n parasite couples and the original signature is preserved although lightly incremented.

References

[1] Abu-Mostafa, Y. S. and Psaltis, D. (1984). Recognitive aspects of moment invariants. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, **PAMI-6**(6), 698 –706.

[2] Alt, F. L. (1962). Digital pattern recognition by moments. *J. ACM*, **9**(2), 240–258.

[3] Chen, Q.-S., Defrise, M., and Deconinck, F. (1994). Symmetric phase-only matched filtering of fourier-mellin transforms for image registration and recognition. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, **16**(12), 1156 –1168.

[4] Derrode, S. and Ghorbel, F. (2004). Shape analysis and symmetry detection in gray-level objects using the analytical Fourier-Mellin representation. *Signal Process.*, **84**(1), 25–39.

[5] EgmontPetersen, M. (2002). Image processing with neural networks—a review. *Pattern Recognition*, **35**(10), 2279–2301.

[6] Goshtasby, A. (1985). Template matching in rotated images. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, **PAMI-7**(3), 338 –344.

[7] Horn, B. (1987). Closed form solutions of absolute orientation using orthonormal matrices. *J. of the Optical Soc. Am.*, **JOSA-5**(7), 1127–1135.

[8] Hu, M. K. (1962). Visual pattern recognition by moment invariants. *IRE Trans. on Information Theory*, **IT-8**, 179–187.

[9] Shahbazkia, H. and dos Anjos, A. (2009). Quick invariant signature extraction from binary images. In *Proc. IEEE Int Signal Processing and Information Technology (ISSPIT) Symp*, pages 172–177.

[10] Sheng, A. and Arsenault, H. (1986). Experiments on pattern recognition using invariant fourier-mellin descriptors. *Journal of the Optical Society America*, **3**(6), 771–776.

[11] Spirkovska, L. and Reid, M. (1990). Connectivity strategies for higher-order neural networks applied to pattern recognition. In *Neural Networks, 1990., 1990 IJCNN International Joint Conference on*, pages 21 –26 vol.1.

[12] Teh, C.-H. and Chin, R. (1988). On image analysis by the methods of moments. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, **10**(4), 496 –513.