INTEGRATING VISION TOOLS IN ROBOTIC TASKS FOR SIGNATURE ANALYSIS, PART MEASUREMENT AND RECOGNITION

Silvia ANTON[1]

În această lucrare, sunt prezentate aspecte de bază privind analiza şi interpretarea imaginii pentru analiza de semnătură. Prima parte analizează cele mai utile seturi de trăsături ale regiunii şi frontierei pentru descrierea formei; astfel de descriptori permit maparea spaţiului obiect în spaţiul trăsăturilor ce vor fi folosite în recunoaşterea statistică a şabloanelor pentru construirea clusterelor şi identificarea şabloanelor. Trăsăturile standard intrinseci sunt funcţii predefinite global aplicabile oricărei clase de obiecte; ele sunt extinse cu masurări globale şi locale ce sunt aplicate claselor particulare de obiecte. Lucrarea este destinată măsurării şi recunoasterii obiectelor prin analiza de semnătură.

In this paper, the basic aspects concerning image analysis and interpretation for signature analysis are presented. The first part analyses the most useful sets of region- and boundary features for shape description; such descriptors allow mapping the object space into the feature space which will be used in statistical pattern recognition for cluster building and pattern identification. Standard intrinsic features are predefined as global functions applicable to any class of objects; they are extended with local and global measurements which are applied to particular classes of objects. The paper is devoted to object measurement and recognition by signature analysis.

Keywords: robot vision, object recognition, signature analysis, inspection, learning.

1. Introduction

Part measurement and recognition are basic functions in merged Guidance Vision for Robots (GVR) and Automated Visual Inspection (AVI) tasks. Internal features describe a region in terms of its internal characteristics (the pixels comprising the body). An internal representation is selected either for the recognition of simple shapes based on sets of *standard intrinsic features* (number of holes, area, perimeter, compactness, eccentricity, invariant moments, etc) or when the primary interest is on reflectivity properties of the object's surface or by statistical, structural or spectral approaches).

[1] PhD Student, Dept. of Automation and Industrial Informatics, University Politehnica of Bucharest, Romania

Internal scalar transform techniques generate shape descriptors based on the region shape. One of the most frequently used methods is that of moments. The *standard moments* m_{pq} of order $(p+q)$ of an image intensity function $f(x,y)$ are:

$$m_{pq} = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} x^p y^q f(x,y)dxdy \quad p,q = 0,1,2,... \tag{1}$$

A uniqueness theorem states that if $f(x,y)$ is piecewise continuous and has nonzero values only in a finite part of the x_{vis}, y_{vis} plane, moments of all order exist and the moment sequence (m_{pq}) is uniquely determined by $f(x,y)$ [1]. Conversely, (m_{pq}) uniquely determines $f(x,y)$.

In the discrete domain of digital images, equation (1) is transformed and the sum is carried out over the entire sub-image within which the shape of interest lies, to get the standard moments of $(p+q)$ order of an object *O*:

$$m_{pq}(O) = \sum_{X=0}^{N_x}\sum_{Y=0}^{N_y} X^p Y^q f_O(X,Y) \quad p,q = 0,1,2,... \tag{2}$$

where:

- $m_{pq}(O)$ is the moment of $(p+q)$ order of object *O*;
- X, Y are the x, y coordinates of the analysed pixel of object *O*;
- $f_O(X,Y) = \begin{cases} 0, & \text{pixel not belonging to object } O \\ 1, & \text{pixel belonging to object } O \end{cases}$
- N_x, N_y are the maximum values respectively for the X, Y image coordinates, e.g. $N_x = 640, N_y = 480$.

However, because these moments are computed on the basis of the absolute position of the shape, they will vary for a given shape *O* depending on its location. To overcome this drawback, one can use the *central moments* of order $(p+q)$:

$$\mu_{pq}(O) = \sum_{X=0}^{N_x}\sum_{Y=0}^{N_y} (X-\bar{X})^p (Y-\bar{Y})^q f_O(X,Y) \quad p,q = 0,1,2,... \tag{3}$$

where $\bar{X} = m_{10}/m_{00}, \bar{Y} = m_{01}/m_{00}$ are the coordinates of the shape's centroid. Thus, these moments take the centroid of the shape as their reference point and hence are position-invariant.

For binary images of objects O, m_{00} is simply computed as the sum of all pixels within the shape. Assuming that a pixel is one unit area then m_{00} is equivalent to the area of the shape expressed in raw pixels.

If the binary image of the object was coded using the run-length coding technique, let us consider that $r_{i,k}$ is the k^{th} run of the i^{th} line and that the first run in each row is a run of zeros. If there are m_i runs on the i^{th} line, and a total of M lines in the image, the area can be computed as the sum of the run lengths corresponding to ones in the image:

$$Area(O) = m_{00}(O) = \sum_{i=1}^{M} \sum_{k=1}^{m_i/2} r_{i,2k} \tag{4}$$

Note that the sum is over the even runs only.

Similarly, m_{10} and m_{01} are effectively obtained respectively by the summation of all the x-coordinates and y-coordinates of pixels in the shape.

The central moments up to order three are given as expressed in (5):

$$\begin{array}{ll} \mu_{00} = m_{00} & \mu_{11} = m_{11} - \bar{Y}m_{10} \\ \mu_{10} = 0 & \mu_{30} = m_{30} - 3\bar{X}m_{20} + 2\bar{X}^2 m_{10} \\ \mu_{01} = 0 & \mu_{03} = m_{03} - 3\bar{Y}m_{02} + 2\bar{Y}^2 m_{01} \\ \mu_{20} = m_{20} - \bar{X}m_{10} & \mu_{12} = m_{12} - 2\bar{Y}m_{11} - \bar{X}m_{02} + 2\bar{Y}^2 m_{10} \\ \mu_{02} = m_{02} - \bar{Y}m_{01} & \mu_{21} = m_{21} - 2\bar{X}m_{11} - \bar{Y}m_{20} + 2\bar{X}^2 m_{01} \end{array} \tag{5}$$

The central moments can be normalized, defining a set of normalized central moments, η_{pq}, and having the expression $\eta_{pq}(O) = \mu_{pq}(O) / \mu_{00}(O)^k$, where $k = (p+q)/2+1,\ p+q = 2, 3, \ldots$.

Moment invariants are preferred for shape description as they generate values which are invariant to translation, rotation, and scale changes. Equation (6) describes a set of seven *invariant moments* which are derived from the second and third normalized central moments.

$$\phi_1 = \eta_{20} + \eta_{02};\ \phi_2 = (\eta_{20} - \eta_{02})^2 + 4\eta_{11}{}^2$$

$$\phi_3 = (\eta_{30} - 3\eta_{12})^2 + (3\eta_{21} - \eta_{03})^2;\ \phi_4 = (\eta_{30} + \eta_{12})^2 + (\eta_{21} + \eta_{03})^2$$

$$\begin{aligned} \phi_5 = {} & (\eta_{30} - 3\eta_{12})(\eta_{30} + \eta_{12})[(\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2] + \\ & (3\eta_{21} - \eta_{03})(\eta_{21} + \eta_{03})[3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2] \end{aligned}$$

$$\begin{aligned}\phi_6 &= (\eta_{20} - \eta_{02})[(\eta_{30} + \eta_{12}) - (\eta_{21} + \eta_{03})^2] + \\ &\quad 4\eta_{11}(\eta_{30} + \eta_{12})(\eta_{21} + \eta_{03})\end{aligned} \tag{6}$$

$$\begin{aligned}\phi_7 &= (3\eta_{21} - \eta_{03})(\eta_{30} + \eta_{12})[(\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2] - \\ &\quad (3\eta_{12} - \eta_{30})(\eta_{21} + \eta_{03})[3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2]\end{aligned}$$

Shape descriptors based on moment invariants convey significant information for simple objects but fail to do so for complicated ones. Since we are dealing with internal scalar transform descriptors, it would seem that these moments can only be generated from the entire region. However, they can also be generated from the boundary of the object by exploiting the theorems of Stoke or Green, both of which relate the integral over an area to an integral around its contour [2].

The simplest internal scalar feature of a region to be identified in robot-vision tasks is its *area*, defined as the number of pixels contained within its boundary. *Compactness* and *roundness* can also be considered as scalar region descriptors, as their formulae contain the blob's area. Compactness is a dimensionless feature and thus is invariant to scale changes.

The *principal axes* of a region are the eigenvectors of the covariance matrix obtained by using the pixels within the region as random variables.

One solution adopted frequently to overcome this difficulty is to use as an internal scalar transform descriptor the ratio of the large to the small eigenvalue. Other simple internal scalar descriptors based on the region's area are:

- The *ratio of the areas of the original blob to that of its convex hull.*
- The *ratio of the area of the original blob to that of its circumcircle.*
- The *ratio of the area of the original shape to the area of the minimal bounding rectangle*. This is a measure of rectangularity and is maximized for perfectly rectangular shapes.
- The *ratio of the area of the original blob to the square of the total limb-length of its skeleton*.

Topological properties are used for global descriptions of regions. Such properties are not affected by any deformation (e.g. stretching). Note that, as stretching affects distances, topological properties do not depend on the notion of distance or any properties implicitly based on the concept of distance measure. A widely used topological descriptor is the *number of holes in the region*, which is not affected by stretching or rotation transformations [3].

The number of holes H and connected components C in a region can be used to define another topological feature – the *Euler number* E of a region:

$$E = C - H \tag{7}$$

Recall that a connected component of a set is a subset of maximal size such that any two of its points can be joined by a connected curve lying entirely within the subset. Fig. 1 exemplifies the above defined topological descriptors for the blob image of a carburettor flange:

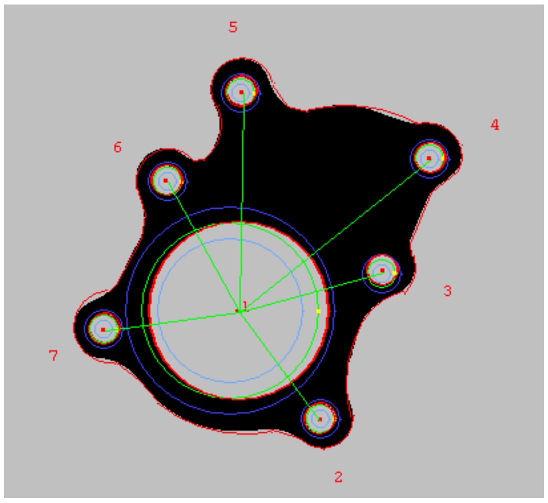

Fig. 1. Topological descriptors of a region: number of holes, $H = 7$, number of connected components, $C = 1$, and Euler number, $E = -6$.

2. Space Domain Descriptors Of Boundaries: The Signatures

External space domain features describe the spatial organization of the object's boundary. One frequently used technique is the use of syntactic descriptors of *boundary primitives*, e.g. atomic edges (lines and arcs), and corners. Thus, the list of shape descriptors (or string of primitive shapes) must follow given rules: the *shape syntax* or grammar.

Signatures are 1-D functional representations of boundaries and may be generated in various ways, for example as polar radii signatures or linear offset signatures. Regardless of how a signature is generated, however, the main idea approached in the present research devoted to real-time visual analysis of parts handled by robots is to reduce the boundary representation to a 1-D function, which is easier to describe than the original 2-D boundary [4], [5].

A *polar radii signature*, encoding the distance from the shape centroid to the shape boundary as a function of angle θ, is shown in Fig. 2.

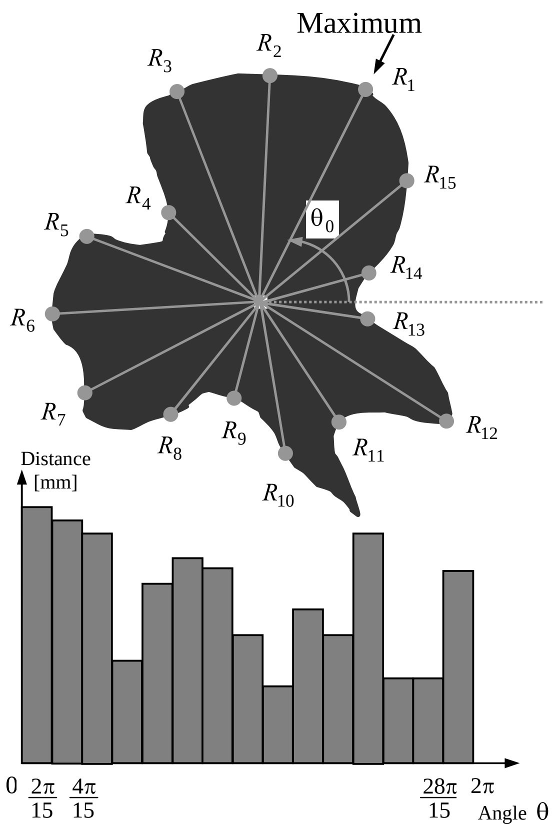

Fig. 2 Angular radii signature of a shape. A 15-element vector $[R_1, R_2,..., R_{15}]$ is defined, where $R_i, i = 1,...,15$ is the distance from the centroid to the edge of the blob, measured at an angle of $(\theta_0 + 24i)$ degrees, and θ_0 is the orientation derived from the greatest radius, R_1.

Such polar radii signatures are invariant to translation, but they do depend on rotation and scaling. To rend such signatures invariant to rotation, there must be found a method to select the same starting point to generate the signature, regardless of the shape's orientation. One possibility is to choose the starting point as the point on the boundary farthest from the blob's centroid, but only if this point is unique and independent of rotational aberrations for each class of objects of interest. Another solution is to select the starting point on the axis of least inertia farthest from the centroid. This method requires more computation, but is more rugged because the direction of the major eigen axis is determined from the covariance matrix, which is based on all boundary points.

Based on the assumption of uniformity in scaling with respect to both axes and that sampling is taken at equal intervals of θ, changes in size of a shape result in changes in the amplitude values of the corresponding signature. One simple way to normalize for the result is to scale all functions so that they span the same range of values, e.g. $[0,1]$. The advantage of this method is simplicity, but the potential drawback is that scaling of the entire function depends on only two values: the minimum and the maximum. If the shapes are noisy, this dependence can be a source of error from one object class instance to the other.

A more robust approach is to divide each sample by the variance of the signature, assuming that the variance is greater than a residual value and hence does not create computational difficulties. Use of the variance yields a variable scaling factor that is inversely proportional to changes in the shape's size.

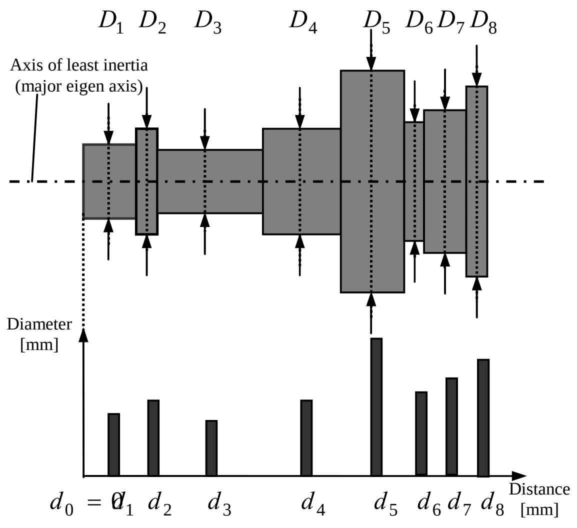
D_1 D_2 D_3 D_4 D_5 D_6 D_7 D_8
Axis of least inertia
(major eigen axis)
Diameter
[mm]
Distance
[mm]
$d_0 = 0$ d_1 d_2 d_3 d_4 d_5 d_6 d_7 d_8

Fig. 3 Linear offset signature of a lathe-turned shape. An 8-element vector $[D_1, D_2, ..., D_8]$ is defined, where $D_i, i = 1,...,8$ is the twice the distance from the minimum inertia axis to the edge of the blob, measured respectively at $d_i, i = 1,...,8$ mm from the "small lateral" edge of the shape.

A linear offset signature, encoding the distance from the axis of least inertia to the shape boundary as a function of distance *d*, is also a space descriptor of the contour. The shape in Fig. 3 has contour segments parallel to its major eigen axis.

It can be observed that in this case sampling is not taken at equal intervals of d, i.e. $d_i - d_{i-1} \neq \text{const}, \ i = 1, ..., 8$.

External space domain descriptors based on signatures are generally simple, and require reduced storage capability.

More complex space domain descriptors are often based on the Fourier series expansion of a periodic function derived from the boundary. For example, the boundary could be traversed at the angle plotted between a line tangent to the boundary and a reference line as a function of position along the boundary.

Consider, for example, the shape depicted in Fig. 4. The rotation angle θ of the tangent at the boundary of the object varies between 0 and 2π radians as the boundary is traversed. In particular, θ will vary with the distance s around the perimeter and can be expressed as a function, $\theta(s)$, called *slope of the boundary*.

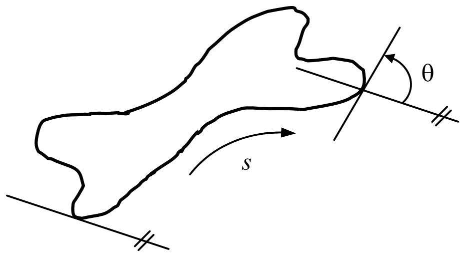

Fig. 4 The slope is obtained by rotation of tangent to the boundary of the shape.

If L is the length of the boundary of the shape, $\theta(0)=0$ and $\theta(L)=-2\pi$. However, the function $\theta(s)$ is not periodic, and consequently it cannot be expressed in terms of a Fourier series expansion. An alternative formulation, suggested by Zhan and Roskies [6], defines a new function, $\phi(t)$:

$$\phi(t)=\theta(\frac{Lt}{2\pi})+t \tag{8}$$

Now, $\phi(0)=\phi(2\pi)=0$, and the function $\phi(t)$ is invariant to scaling, translation and rotation of the shape; hence, the low-order coefficients of its Fourier expansion can be used as features for translation, rotation, and scaling in shape recognition.

A variation of this approach is to use the so-called *slope density function* as a signature. This function is simply a histogram of tangent-angle values. As a histogram is a measure of concentration of values, the slope density function highlights sections of the boundary with constant tangent angles (straight or nearly straight segments) and has deep valleys in sections producing rapidly varying angles (corners or sharp inflexions).

The *curvature* is defined as the rate of change of the slope. In general, obtaining reliable measures of curvature at a point in a digital boundary is difficult because the boundaries tend to be locally "ragged". A solution consists into using the difference between the slopes of adjacent atomic boundary segments (e.g. represented as straight lines) as a descriptor of curvature at the point of intersection of the segments.

3. Conclusions

The research presented in this paper was directed towards integrating a set of efficient, high-speed vision tools: *Windows Region of Interest* (WROI), point-, line-, and arc *finders*, and linear and circular *rulers* into an algorithm of interactive signature analysis of classes of mechanical parts tracked by robots in a flexible production line. To check the geometry and identify parts using the signature, you must follow the steps:

1. Train an object

The object must represent very well its class – it must be "perfect". The object is placed in the plane of view and then the program which computes the signature is executed; the position and orientation of the object is changed and the procedure is repeated for a few times.

The user must specify for the first sample the starting point, the distances between each ruler and the length of each ruler. For the example in Fig. 5 of a **linear offset signature**, one can see that the distances between rulers (measurements) are user definable.

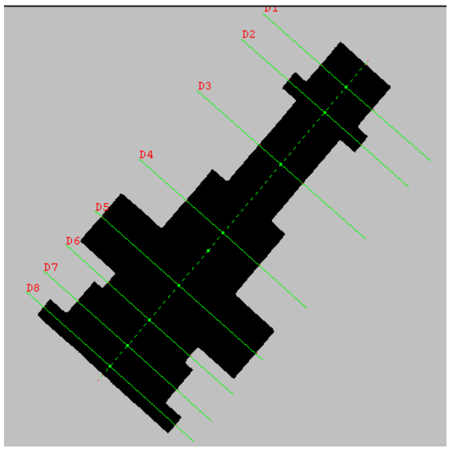

Fig. 5 The linear offset signature of a lathe-turned shape

If we want to compute a **polar signature** the user must specify the starting point and the angle between each measure.

During the training session, the user can mark some edges (linear or circular) of particular interest that will be further searched and analysed in the recognition phase. For example if we want to verify if a linear edge is inclined at a certain angle with respect to the part's Minimal Inertia Axis (MIA), the start and the end point of this edge will be marked with the mouse of the IBM PC terminal of the robot-vision system. In a similar way, if a circular edge must be selected, we will mark the start point, the end point and a third point on that arc-shaped edge.

The program computes one type of signature according to the object's class. The class is automatically defined by the program from the numerical values of a computed set of standard scalar internal descriptors: compactness, eccentricity, roundness, invariant moments, number of bays, a.o.

After the training the object has associated a class, a signature, a name and two parameters: the tolerance and the percentage of verification.

2. <u>Setting the parameters used for recognition</u>

- The *tolerance*: each measure of the recognized object must be into a range: (original measure ± the tolerance value). The tolerance can be modified anytime by the user and will be applied at run time by the application program.
- The *percentage of verification*: specifies how many measures can be out of range (100% – every measure must be in the range, 50% – the maximum number of rulers that can be out of range is ½ of the total number). The default value of the percentage of verification proposed by the application is 95%.

3. <u>The recognition stage</u>

The sequence of operations used for measuring and recognition of mechanical parts includes: taking a picture, computation of the class to which the object in the WROI belongs, and finally applying the associated set of vision tools to evaluate the particular signature for all trained objects of this class.

The design of the signature analysis program has been performed using specific vision tools on an Adept Cobra 600 TT robot, equipped with a GP-MF602 Panasonic camera and AVI vision processor.

The length measurements were computed using linear rulers (VRULERI), and checking for the presence of linear and circular edges was based respectively on the finder tools VFIND.ARC and VFIND.LINE [7].

The pseudo-code below summarizes the principle of the interactive learning (developped in this paper) during the training stage and the real time computation process during the recognition stage.

```
i) Training
    1. Picture acquisition
    2. Selecting the object class (from the computed values of internal
       descriptors: compactness, roundness,...)
    3. Suggesting the type of signature analysis:
        3.1. Linear Offset Signature (LOF)
            3.1.1. specify the starting point and the linear offsets
```

3.2. Polar Signature (PS)

3.2.1. specify the starting point and the incremental angle

4. *Specify the particular edges to be verified*
5. *Improve the measurements?*

   5.1. Compute repeatedly only the signature (the position of the object is changed every time)

   5.2. Update the mean value of the signature.
6. *Compute the recognition parameters* (tolerance, percentage of verification) and *name* the learned model.
7. *Display the results and terminate the training sequence*.

ii) Run time measurement and recognition

1. *Picture acquisition*
2. *Identifying the object class* (using the compactness, roundness,... descriptors)
3. *Computing the associated signature analysis* for each class model trained.
4. *Checking the signature against its trained value*, and inspecting the particular edges (if any) using finder and ruler tools
5. *Returning the results* to the AVI program or GVR robot motion planner (the name of the recognized object, or void).
6. *Updating the reports* about inspected and/or manipulated (assembled) parts; sequence terminated.

Fig. 6 and Table 1 show the results obtained for a polar signature of a leaf-shaped object.

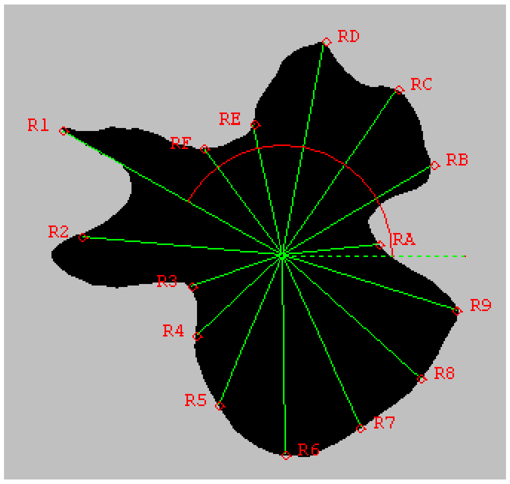

Fig. 6. Computing the polar signature of a blob.

Table 1

Statistical results for the polar radii signature of the leaf-shaped object

Statistics [mm] / Parameter	Min value (min)	Max value (max)	Mean value (*avg*)	Dispersion (*disp*)	Number of ruler tools used
$R1$	68.48	70.46	68.23	1.98	1
$R2$	56.59	58.44	57.02	1.85	1
$R3$	26.68	28.42	27.41	1.74	1
$R4$	32.24	34.03	33.76	1.52	1
$R5$	44.82	45.92	45.42	1.10	1
$R6$	54.07	55.92	54.83	1.85	1
$R7$	51.52	52.76	52.05	1.24	1
$R8$	50.39	51.62	50.98	1.23	1
$R9$	49.15	51.18	49.67	2.03	1
RA	25.41	26.98	26.22	1.57	1
RB	47.41	48.68	47.91	1.27	1
RC	53.71	55.30	54,64	1.59	1
RD	57.79	59.51	58.61	1.72	1
RE	35.69	37.39	36.80	1.70	1
RF	35.42	36.72	36.17	1.30	1

The dispersion was calculated for each parameter $P_i, i = 1,...,10$ as: $disp(P_i) = \max(P_i) - \min(P_i)$, and is expressed in the same units as the parameter (millimetres or degrees). The min/max values are: $\min = \min(P_i)$, $\max = \max(P_i)$. The expression of the mean value $avg = \frac{1}{10}\sum_i P_i$.

R E F E R EN C E S

[1] *Borangiu, Th.*, Intelligent Image Processing in Robotics and Manufacturing, Romanian Academy Press, Bucharest, 2004, pp. 1 – 650.

[2] *Ams, E*. Eine für alles?, Computer & Automation, no. 5, 2002, pp. 22-25.

[3] *Fogel, D.B*. Evolutionary Computation: Toward a New Philosophy of Machine Intelligence, IEEE Press, New York, 1994.

[4] *Borangiu, Th. and L. Calin*. Task Oriented Programming of Adaptive Robots and Integration in Fault–Tolerant Manufacturing Systems, Proc. of the Int. Conf. on Industrial Informatics, Section 4 Robotics, Lille, 1996, pp. 221-226.

[5] *Camps, O.I., L.G. Shapiro and R.M. Harlick*. PREMIO: An overview, Proc. IEEE Workshop on Directions in Automated CAD-Based Vision, 1991, pp. 11-21.

[6] *Zhan, C.T. and R.Z. Roskies*. Fourier descriptors for plane closed curves, IEEE Trans. Computers, **vol. C-21**, 1972, pp. 269-281.

[7] *Adept. AdeptVision User's Guide*, Version 14.0, Part Number 00964-03300, Rev. B, Adept Technology Inc., San Jose, CA, 2001.