OBJECT ORIENTED SOFTWARE FOR MICRO WORK PIECE RECOGNITION IN MICROASSEMBLY

Toledo-Ramírez, G.K., Kussul, E.& Baidyk, T.

Centro de Ciencias Aplicadas y Desarrollo Tecnológico (CCADET), UNAM
Circuito Exterior S/N CP 04510, Ciudad Universitaria, Mexico DF
52-56228602 ext. 1135, Fax 56228626.

Received: September 5th, 2005. Accepted: March 16th, 2006

ABSTRACT

The aim of this article is to describe object oriented software for the automatic micro work piece handling system. The general task of this system is the recognition of work pieces with neural classifier and detection of their positions. Other important functions of the system are work piece styles database administration, work piece database administration for neural classifier training and testing, neural classifier interface between database, user and work piece finder. The software is object oriented and widely commented, that makes its modification, adaptation and improvement easier. Most of the software modules can be used in other research projects. The software was tested on image database. The results of experiments prove its effectiveness in chosen task.

RESUMEN

El propósito de este artículo es describir un software orientado a objetos para el sistema automático de manejo de micro piezas. La tarea general del sistema es reconocer las distintas piezas de trabajo con redes neuronales y sus posiciones. Otras funciones importantes del sistema son el manejo de las bases de datos de estilo de piezas y de piezas para entrenamiento y prueba de la red neuronal, así como la interfaz del clasificador neuronal entre la base de datos, el usuario y el localizador de piezas. El software es orientado a objetos y ha sido ampliamente comentado, esto hace más fácil su modificación, adaptación y mejora. La mayor parte de los módulos de software pueden ser utilizados en otros proyectos de investigación. El software ha sido probado sobre una base de datos de imágenes de piezas. Los resultados de los experimentos prueban la efectividad del software en las tareas descritas.

KEYWORDS: Automatic Handling System, Neural Classifier, Technical Vision, Position Recognition, Microassembly, Object Oriented Software.

1. INTRODUCTION

The miniaturization of products coupled with development of new technologies [1, 2] gives us possibility to develop equipment and instruments of small dimensions. Our aim is to create fully automated desktop microfactory. We consider sequential generations of microfactories. Microfactories of each generation are made by means of microfactories of previous generations and have smaller corresponding dimensions. This approach is called "microequipment technology" (MET) [3, 4].

Automation of micromanufacturing and microassembly processes is the fundamental task in order to achieve good performance of corresponding microequipment [5, 6]. It is also important to apply computer vision methods to the

processes of micromanufacturing and microassembly. One of the essential parts of the automated microfactory that we propose is a fully automated work piece handling system. The goal of this system is to recognize one randomly located work piece in work area, to define its coordinates, to take it with manipulator and to transport it to another place. This system has to be flexible. This means, that the system must be able to handle work pieces of different dimensions and shapes. The scheme of the proposed system is presented in Fig. 1.

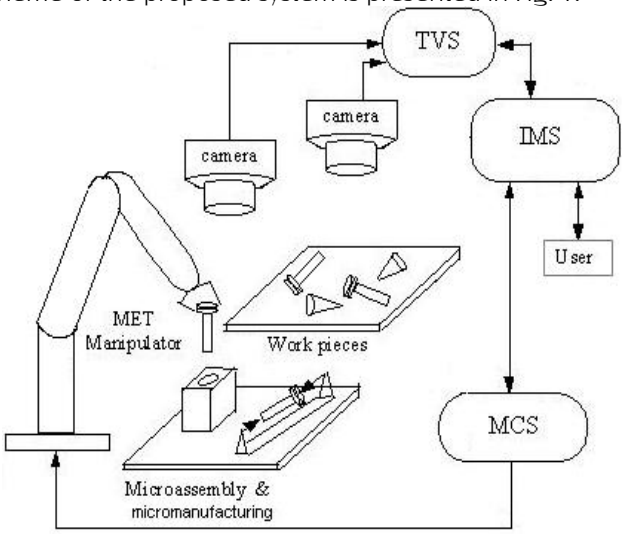

Figure 1: Scheme of the proposed automatic work pieces handling system.

This system consists of two cameras, one manipulator and three subsystems: manipulator control subsystem (MCS), technical vision subsystem (TVS) and intelligent manipulation 2

subsystem (IMS). The manipulator has six degrees of freedom and special characteristics to handle work pieces in micro dimensions. We use general purpose, low cost, commercial CD cameras. They produce input images for the TVS. The main purpose of this subsystem is to process the data from the cameras and to recognize work pieces and their positions. The IMS controls the whole system. This means that it gives the instructions to MCS, processes the data from the TVS and coordinates cooperation between system and user. The user can be a person or another system.

We began to work with the system that contains one camera and the TVS. TVS is based on the method of micro work piece recognition and their positions detection. For this purpose we adapted the neural classifier LIRA grey scale [7]. Position detection gives us the coordinates of recognized work pieces. In work piece recognition task we obtained the recognition rate of more than 90%.

The paper is organized as follows: In Section 2 we describe our TVS. In Section 3 we describe the neural classifier LIRA, processes of training and recognition. In Section 4 we describe the developed software and in Section 5 we describe image databases that were used in the experiments. The experiments and results are presented in the Section 6. Discussion and conclusion are presented in Sections 7 and 8 correspondingly.

2. TECHNICAL VISION SUBSYSTEM (TVS)

The description of the object recognition problem and methods that are widely used to solve it are presented in [8–13]. Neural networks are used for object recognition. Such characteristics of neural networks as adaptive learning, flexibility and real time recognition are very useful in object recognition task. There are works on work piece recognition for robot control or manufacturing systems that use neural networks as its principal recognition technique. Detter and Radjenovic [14] use preprocessed image as input to the neural classifier. The best recognition rate obtained was 97.7% for three classes of objects. The time demands were 1.2 seconds for preprocessing and 0.4 for recognition. Mitzias and Mertzios [15] use in their work a multi neural classifier, 3 i. e. three neural networks working in parallel. A different preprocessing procedure was used for each one of the classifiers i.e. each classifier worked with different characteristics of the image. To examine this method they used nine irregular flat objects with a

hole. This approach requires a lot of time and computer resources to satisfy real time requirement for the recognition task.

The system for microassembly has been proposed by Mardanov et al [16]. The proposed microassembly station has specific logical modules and common information paths in order to be integrated easily into bigger automated assembly system.

We use neural classifier LIRA which has been tested in object recognition task with good results [17, 18]. The aim of the TVS (Fig 1) is to recognize microobjects and their positions. It has three subsystems: interface, object finder that searches for certain work pieces and defines their position and orientation, and the camera controller that operates the camera and its data flow.

In Fig. 2 the work piece object and its coordinates are presented, where x, y and θ are the coordinates and the orientation angle of the work piece. It is important to find a work piece on the image, its coordinates and orientation in acceptable time.

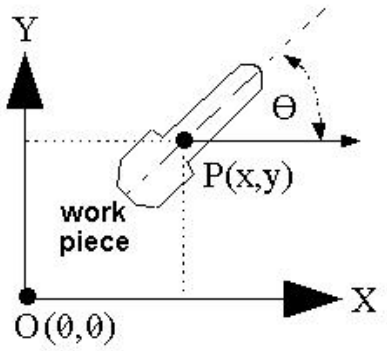

Figure 2: Example of the work piece object with coordinates (x, y, θ) in rectangular coordinate system.

To test TVS we used different types of work pieces that are shown in Fig. 3. Sometimes these work pieces have dirty surfaces, heterogeneous or low illumination; sometimes they are surrounded by shadows, all that makes the recognition task more complicated. Every work piece has to be recognized in a set of random distributed work pieces. We will describe the 4 work piece image databases that we used. In the next section we will describe the proposed method of image recognition.

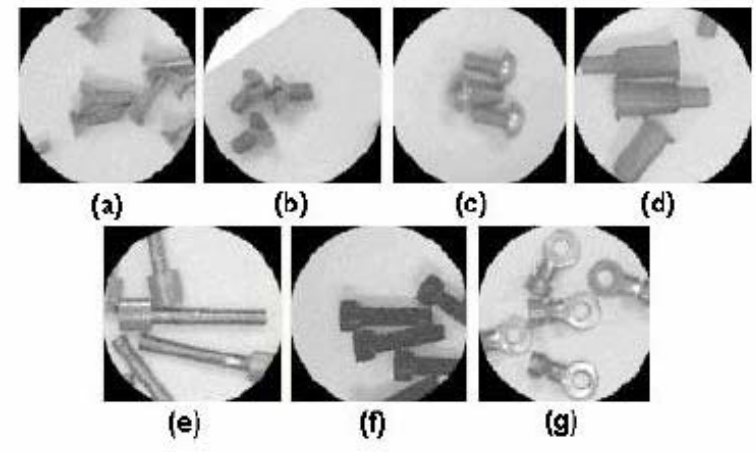

Figure 3: Seven different types of work pieces: a) tube base, b) cone head screw, c) round head screw, d) cone, e) stator axis, f) allen screw, g) wire terminal.

3. NEURAL CLASSIFIER

We use a neural classifier LIRA (LImited Receptive Area) based on modified Rosenblatt Perceptron [19]. This classifier was proposed and tested; it showed good results in handwritten digit recognition task [20]. We want to adapt this neural classifier to resolve our task. Let us describe the LIRA structure.

The LIRA neural classifier is an artificial neural network with four layers (Fig. 4):

- Input layer (S) $S = s_1, s_2, ..., s_k$
- Layer of groups (I) $Group = group_1, ..., group_N$
- Associative layer (A) $A = a_1, a_2, ..., a_N$
- Output layer (R) $Y = y_1, y_2, ..., y_m$

S layer corresponds to the input image to be classified. In this layer, called retina, each neuron corresponds to the brightness of each pixel of the image to be processed; therefore the output range of these neurons is $[0,B]$, where 0 stands for the null brightness (black) and B stands for the highest brightness. This layer has $W \cdot H$ neurons, where W and H are respectively the width and the height of the image to be classified. I layer contains N neuron groups, each group i has p ON-neurons and q OFF-neurons. Each ON-neuron is active if $x_{ij} > T_{ONij}$, each OFF-neuron is active if $x_{ij} < T_{OFFij}$, where x_{ij} is the input of the corresponding neuron, T_{ONij} is the ON-neuron threshold and T_{OFFij} is the OFF-neuron threshold. Each neuron threshold is randomly selected within $[0, B \cdot \eta]$, where η is an experimental constant of the classifier to be selected from $(0, 1]$. The neurons of the group are randomly connected with the neurons of S-layer located in a rectangular window $w \cdot h$ defined in S layer (Fig. 4). Values dx and dy are selected randomly from $[0,W - w]$ and $[0,H - h]$ respectively. Parameters w and h are very important for performance of the neural classifier, and are chosen experimentally. Connections between layers S and I and thresholds of I layer does not change during the training process.

The A layer contains N neurons, each neuron has $p + q$ inputs connected to the outputs of corresponding group. The A-layer neuron is active if, and only if all its inputs are active, the neuron output equals 1 if it is active and equals 0 if it is not, this type of neurons is called AND-neurons. Connections between layers I and A does not change during the training process. All A layer neurons are connected to every R layer neuron. Weights of these connections are modified during the training process. The output of R layer adding neuron I is calculated with the following equation:

$$y_i = \sum_{j=0}^{N} w_{ji} \cdot a_j \qquad (1)$$

where w_{ji} is the connection weight between A layer neuron j and R layer neuron i and a_j is the output of the neuron j from the A layer.

When an image is set to the classifiers input we calculate the activity of A layer neurons and present this activity as a binary vector $\vec{A}$. This calculation we term "image coding" and the vector $\vec{A}$ we term "image code". We store the image codes of training set not to repeat the coding process in each training cycle. Training process is described in detail in our publications [17, 18].

The classifier inputs are randomly distributed through the I layer. For example, if we have input image with dimension of 150 • 150 pixels, each one of the 22500 neurons of the S layer can be randomly used by the 175000 • 7 = 1225000 neurons of the I layer, where 175000 is the number of groups and 7 is the number of neurons in each group. To

ensure these pixels are randomly distributed through the image, our software builds the histogram representation of this distribution (Fig. 5).

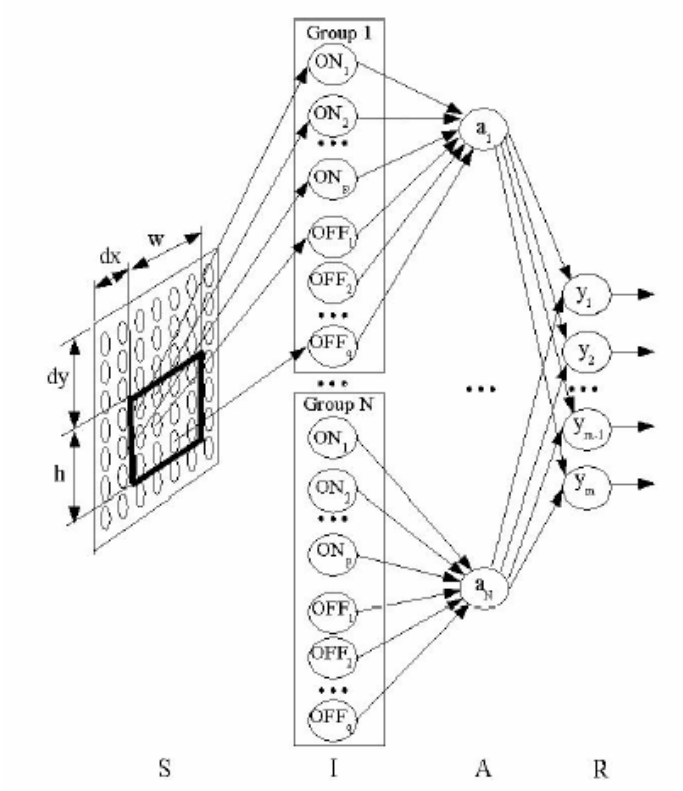

Figure 4: LIRA grey scale neural classifier structure.

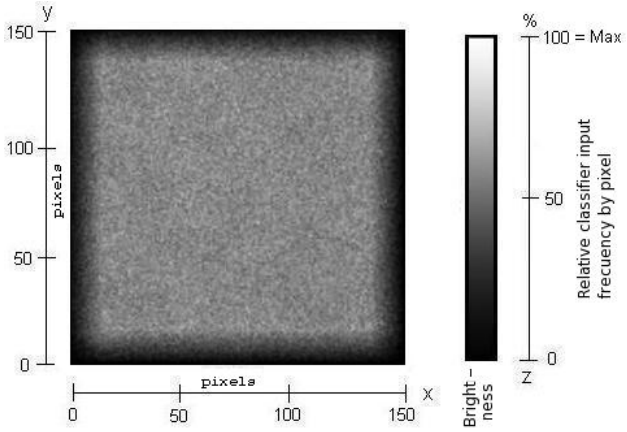

Figure 5: Histogram representation of the neural network input distribution.

The histogram is in 3D, each histogram pixels brightness corresponds to the number of times the each input image pixel appears in the I layer of the classifier. Black pixel stands for no use and white pixel stands for maximal use. For our histogram, maximal pixel use is 102 times. We can see that the distribution of inputs in our classifier is good, noting that distribution at edges is worth, but it is not important due to the fact that work pieces to be trained are situated in the center of the image samples. The training image set can be enlarged by adding new images obtained applying distortions to the original images. In our work, we applied rotational distortions. Each image was rotated in

clockwise and counter clockwise directions forming pairs of new images $\pm I^{\circ}$, $2 \cdot I^{\circ}$, ..., $\mathrm{P} \cdot I^{\circ}$, where I is the rotation step for each pair and P is the number of pairs.

4. SOFTWARE

An object oriented software called OptikRna that we developed realizes LIRA neural classifier and creates databases for classifiers training and testing. Structure of OptikRna is presented in Fig. 6. This software has three principal modules: Optik, Rna (from Spanish abbreviation of artificial neural network) and Rna interface. Optik functions help the user to create work piece image database for the neural classifiers training and testing, to define characteristics of work pieces and to do some basic preprocessing procedures. Rna module realizes the LIRA neural classifier. The Rna module functions and the cooperation between user and databases are performed by Rna interface module. The Rna interface module searches for certain work piece on the image and defines its position.

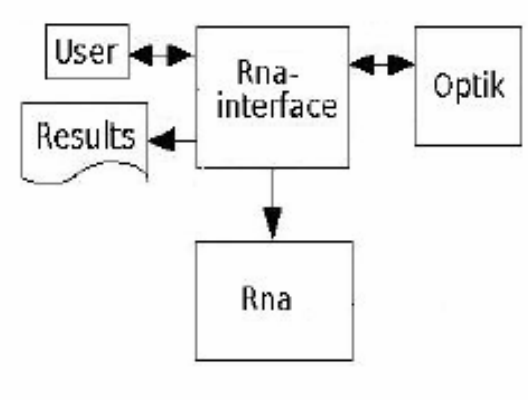

Figure 6: Block diagram of the OptikRNA software.

The user defines the properties like height or width for each type of work piece in Optik module. This information is stored by the system. Source image contains several random located work pieces. The user puts a mark on each work piece approximately in the centre of its major axis. Then Optik module derives and normalizes image samples of each work piece.

These normalized image samples are stored creating in such manner the image database. The object oriented OptikRna software has all the advantages of Object Oriented Programming (OOP). We used the C++ programming language and Borland® Integrated Development Environment (IDE). Some predefined Borland® objects were used for image manipulation and creation of Graphical User Interface (GUI).

The results of OptikRna software testing are presented at the end of this article.

4.1. OptikRna software

Our software is divided in two modules called Optik and REDNEUART. We have made our software following the OOP paradigms. We had implemented other programming techniques in order to make the software maintenance, i.e. improvements and additions, and its interaction with other software. These techniques are separation of GUI from the rest of the code and code comments.

4.2. Neural classifier implementation

Our neural classifier software is based on integer numbers operations in order to avoid large time effort that is necessary for floating point operations. The most general classes were made, that means that the same software modules can be used to create different neural network topologies. The user interface was made specially for LIRA neural classifier. It is not hard to use this software modules to construct other types of neural networks or make changes to the current topology, e. g. add more layers.

The most important classes created for artificial neural network realization were: Dentrita, Neuron, NeuralSet and Rna.

The Dentrita class is a basic one. It has only two attributes, stimulus and weight. Dentrita instances are used widely by neuron objects in order to create fully functional neurons. In Fig. 7 the Neuron class and its derived classes are shown. The neuron types used in LIRA classifier are ON, OFF, AND and adding neurons. The ON and OFF neurons are one input (OI) neurons and the rest are several input (SI) neurons. The Neuron class is the fundamental part of the neural network software.

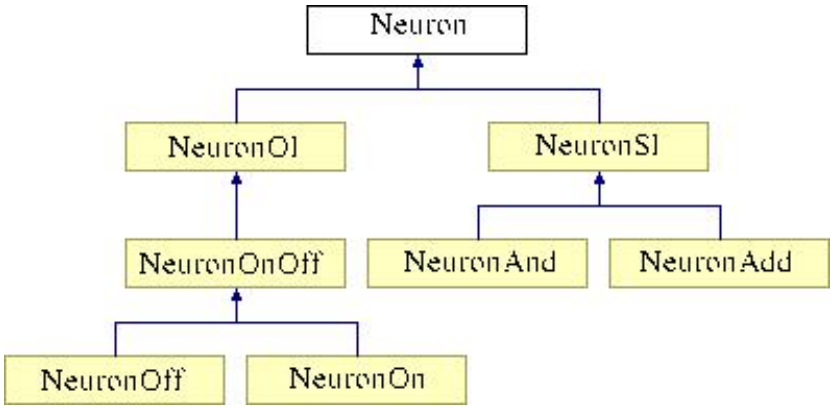

Figure 7: Neuron class and its derived classes.

In order to construct neuron sets a general NeuralSet class was created. Rna is the class that combines all mentioned classes. This class we use to construct a LIRA neural network. Examples of the parameters are the number of neurons or groups in each layer, the number of ON and OFF neurons in each group, the input vector and the output classes. The Rna object contains information about itself. It is possible to store the complete neural network including all its internal parameters, to load a previously stored neural network and to perform training and recognition processes.

The neural classifier is controlled by an object called Rna-Interface. That interrelates the user by means the GUI with the databases used for training and testing of the classifier.

4.3. OptikRna Graphical User Interfaces (GUIs)

We have developed two GUIs, one for image database creating called Optik and other for the artificial neural network called Rna. With the Optik GUI user defines the properties of work pieces and creates image sample databases. Rna GUI gives possibility to control with the neural classifier, to work with the created databases and to use the work piece finder.

An initial image has several randomly located work pieces. The user using developed software tools marks each work piece at the middle of the edge taken as the work pieces base point and defines its orientation (Fig. 8). Before making marks, user has to define the type of work piece to be marked and software calculates the mm-pixel factor. After the mark is put on the work piece on the initial image, using the length of the work piece derived from its type and the mm-pixel factor software calculates the coordinates of the centre of work piece.

With these coordinates, software extracts the image sample of each work piece. Before saving these samples are modified by applying a circular window algorithm to make future image rotation easier. The work piece is rotated to the same orientation angle but with opposite sign to fix their orientation. The work piece type is used to name the sample files. The type name is added to an ordinal number in order to generate the respective sample file name. Original centre coordinates and orientation are stored too.

Because of the fact that marks on the initial images were put by user the image samples in database are not oriented or centred perfectly. This is not a problem because classifier is able to perform the task of recognition with imperfect samples in case of large training set. User works with the neural network software module that realizes LIRA classifier

by means of Rna GUI (Fig. 9). With help of this GUI user can create the neural classifier and the available work piece image databases, train and test the classifier and use the work piece finder. The Rna GUI with a trained classifier loaded is shown in Fig. 9. Information about the classifier is shown in the text box on the left. Results are given at the text box on the right: the obtained recognition and the filenames of the samples from the processed database that system could not have recognized.

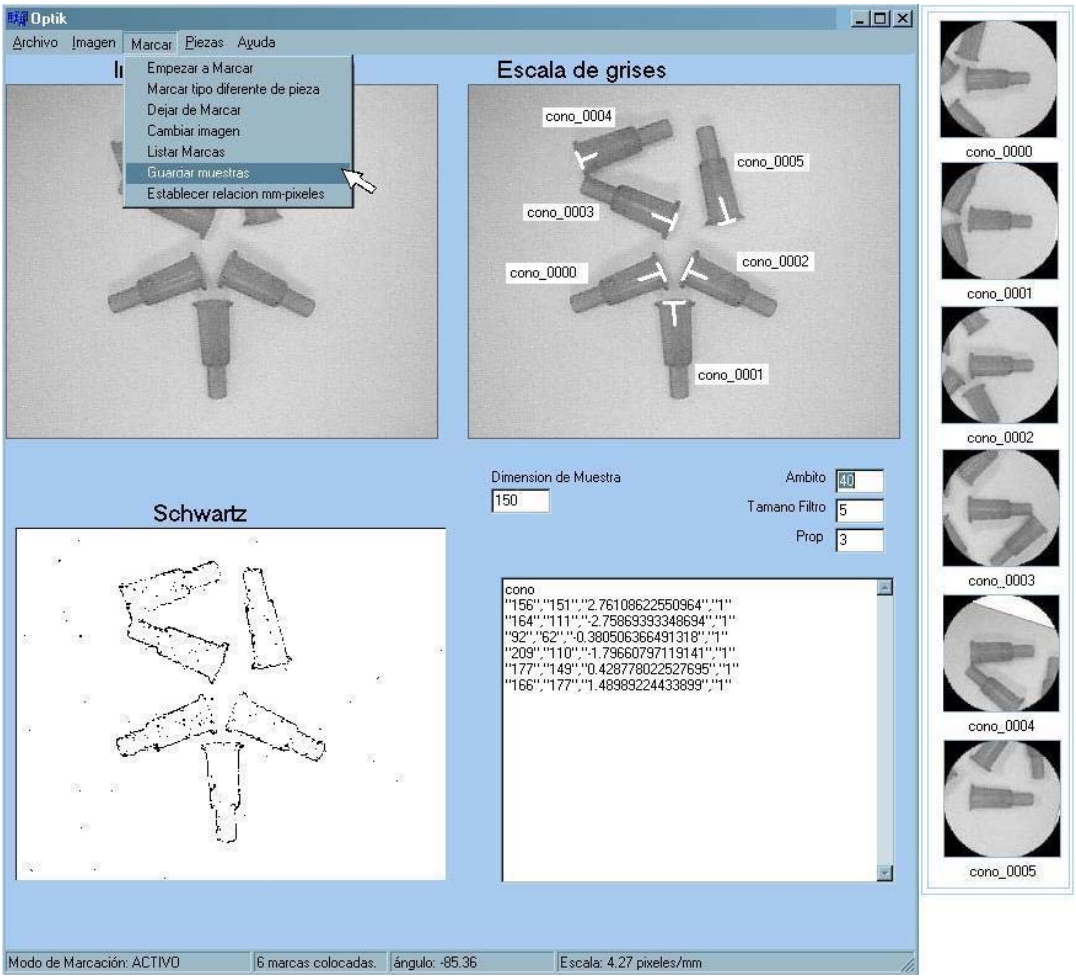

Figure 8: Optik GUI, which allows to mark and extract image samples and contours (samples extracted from the current image are shown on the right.)

In the centre of Fig. 10 is shown a work piece initial image which has been processed by the work piece finder. A trained and tested neural classifier was previously loaded. Also in this image are shown a mark on a work piece found by the finder and the position and orientation associated with this found work piece in the text box on the right.

5. DATABASE

We use two databases in our experiments. The first and smaller database, that contains 150 images was used for initial experiments with LIRA classifier to tune parameter values.

The second and larger database, that contains 320 images was used to verify classifier parameter values found with the first database and for practical recognition and experiments with position finder.

For our experiments, we chose seven different types of work pieces (Fig.3). Chosen work pieces mostly have different sizes (between 28.8 and 4.2 mm.) and shapes. The work piece geometry has circular parts.

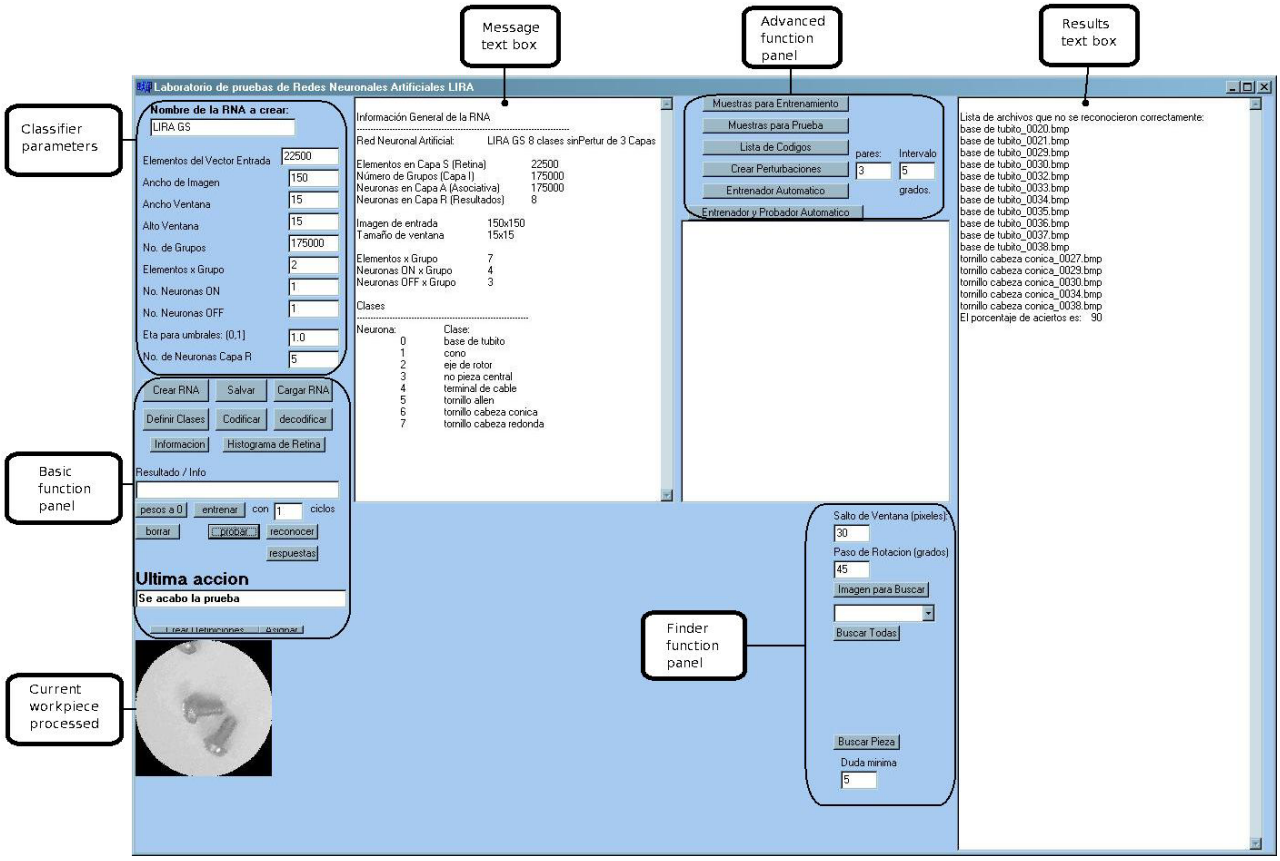

Figure 9: Rna GUI.

There are three classes of work pieces which are very similar (Fig. 3 a, b, c.). Sometimes they have dirty surfaces that corresponds to real conditions of micromechanical work areas.

Both databases were created by means of specially designed software. This software was applied to several initial grey scale images that contained large set of work pieces. All these images were taken by a low resolution CCD camera. A top view of work pieces without special illuminance conditions was used. The user have to mark the centre of the largest axis of all work pieces on the initial image. Then software extracts and normalizes images for database creation. Normalization means that every database image is grey scale with 256 levels, has fixed size of 150 x 150 pixels, work pieces on the image are centred and have fixed orientation (their mayor symmetry axis coincides with axis X). A circular window algorithm was applied to the images to make image rotation easier. Many work pieces on the initial images are heaped up, so the database images can contain parts of some other work pieces in addition to the central one.

The first database contains 30 images for each of five different classes of work piece, 15 images for the training set and other 15 images for the test set. First four classes correspond to four different types of work pieces and the fifth class corresponds to the absence of work piece in the centre of the image. The second database contains 40 images for each of the eight different classes of work pieces, 20 images for the training set and 20 images for the test set. First seven classes correspond to seven different types of work pieces and the eighth class corresponds to the absence of work piece in the centre of the image. The images for the training and test set were randomly selected. The training set of images can be enlarged with distortions applied to the original set. The images have low resolution, different brightnesses and some of them have shadows. In addition, sometimes a background is not homogeneous. Such characteristics represent real conditions and make the recognition problem more complicated.

We trained and tested the classifier with normalized images from our databases. Due to random selection of work pieces from database for train and test sets the obtained results are statistically reliable.

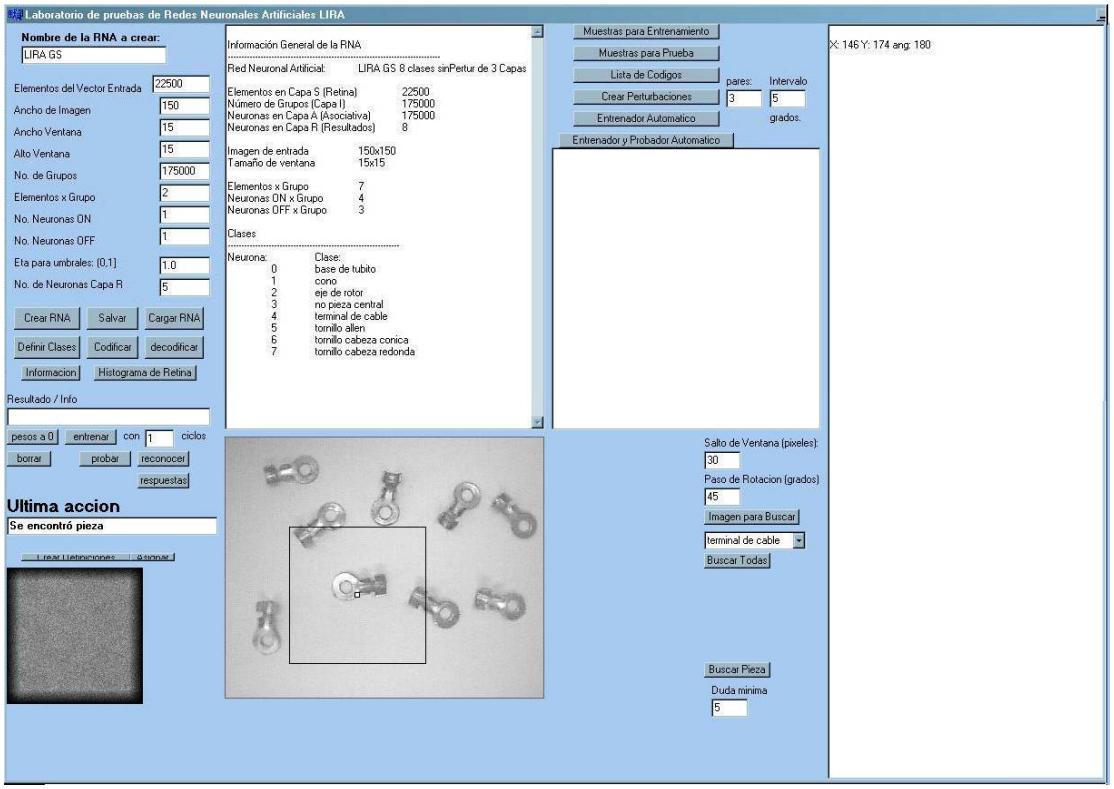

Figure 10: Rna GUI. Workpiece finder based on the previously loaded neural classifier.

Figure 11: Optik GUI. Workpiece characteristics database dialog window.

To improve work piece handling an automated work piece handing system should have general information about work piece types. For this purpose we had integrated in our system a work piece type database manager (Fig. 11). This is a simple software module that allows to create, save and load work piece image databases. In the current version of Optik software work piece database includes general information like name, description, material, geometric information like length (defined as the largest dimension of the work piece), width (the second largest one), height (the third one), weight and volume. Each work piece can be edited in order to modify the data.

The information stored in the work piece image database can be used by several system processes in order to manage objects with better reliability. For example, it is very important to know the material and dimensions of the work piece to be handled to calculate the best parameters of the manipulator that is going to move the work piece in the work area.

6. EXPERIMENTS AND RESULTS

We have tuned the classifier parameters with several experiments on first database. For the LIRA neural classifier we have found the optimal parameter values (for example, w and h for LIRA window, number of training cycles and number of neurons of $\mathbf{A}$ layer) that gave us the best recognition rate. Several experiments were made for this purpose. We did the similar experiments with contour images of work pieces too. Other experiments were made with the second database to recognize the position of the work piece.

Table I: The experiments for the parameter tuning (database I).

Number of experiments:	1	2	**3**	4	5	6	7	8
Window size ($w \cdot h$)	12 · 12	15 · 15	**15 · 15**	10 · 10	13 · 13	17 · 17	15 · 15	10 · 10
Experimental classifier constant (η)	0.8	1.0	**1.0**	1.0	1.0	1.0	1.0	1.0
Number of neurons in layer A (N) (thousands)	175	175	**175**	175	175	175	200	200
Number of ON-Neurons by group (p)	3	3	**4**	4	4	5	4	4
Number of OFF-Neurons by group (q)	4	4	**3**	3	3	3	3	3
Active neurons (%)	0.06	0.09	**0.16**	0.12	0.13	0.08	0.17	0.15
Recognition rate (%)	68	89	**94**	93	85	88	89	89

Table II: Experiments with distortions.

Experiment number	1	2	3	4
Distortions number	6	10	10	16
Step (degrees)	5	10	3	3
Distortions angles	±5, 10, 15	±10, 20, 30, 40, 50	±3, 6, 9, 12, 15	± 3, 6, 9, 12, 15, 18, 21, 24
Training cycles	80	10	60	80
Recognition rate (%)	92	68	88	90

We tested several parameter combinations of the classifier. In the first group of experiments we did not use the distortions of training images. The results of this group of experiments are presented in Table I. The best result was obtained in experiment with a window of 15 × 15 pixels, 175 000 neurons in A layer. In this experiment we obtained the correct recognition rate of 94%. We verified that the recognition rate for a given set of parameters converges with a certain number of training cycles. For experiments of this serie the best result was obtained with 40 training cycles. The training time for 75 images in the experiments of this serie was less than 1.5 minutes.

The experiments of the second group were made adding distortions to the training images. The purpose was to improve classifier recognition of work pieces with different orientations.

The results of these experiments are presented in Table II. The best result was obtained in experiment where six distortions (+15°, +10°, +5°, −5°,−10°,−15°) were added. The training time in case of 16 distortions was

approximately 16 minutes. The recognition time of any normalized image from any database was less than 0.4 seconds.

We made the same experiments with the images which were preprocessed. We extracted the contours of the images using a filter based algorithm. The LIRA classifier, in this case, demonstrated poor results, only 12% of correct recognition. We suppose that recognition rate in this case can be improved with Binary LIRA classifier [20].

We found the best parameters for the LIRA classifier using the first database. After that we trained the classifier with images from the second database without adding distortions. In this case we obtained the best result of 90%. The images from six classes (round head screw, cone, stator axis, Allen screw, wire terminal and no work piece in the central point of image) where recognized with a recognition rate of 100%. The images from the other two classes (tube base and cone head screw) were recognized with several errors.

6.1. Position recognition

The TVS goal is not only to recognize the work piece but to find the position of this work piece for manipulator handling. The point to be found on a work piece is the centre of its major symmetry axis.

We applied the position algorithm to find certain work piece in a given image after the classifier training with the eight classes from the second database. The algorithm starts with the window $(w \times h)$ moving across the whole image. This movement is made from the centre of the initial image (x_0, y_0) in the form of clockwise snail and serves to find a specific work piece (Fig. 12). The little square is the centre of the window. The vertical and horizontal steps of this movement are $\Delta x,\ \Delta y$ respectively. At each position the centre of the window is rotated by an angular step $\Delta\theta$. For each angular position the system searches for the requested work piece. The window continues to rotate until some work piece is recognized or a complete revolution is made. The window centre moves until some work piece is recognized or image limits are achieved. If a work piece is recognized the system store the coordinates $\Delta x,\ \Delta y,\ \Delta\theta$ and finish the recognition task. The user is able to indicate the window displacements parameters $(\Delta x,\ \Delta y,\ \Delta\theta)$ for investigation goals.

In Fig. 13 we show example of recognized image.

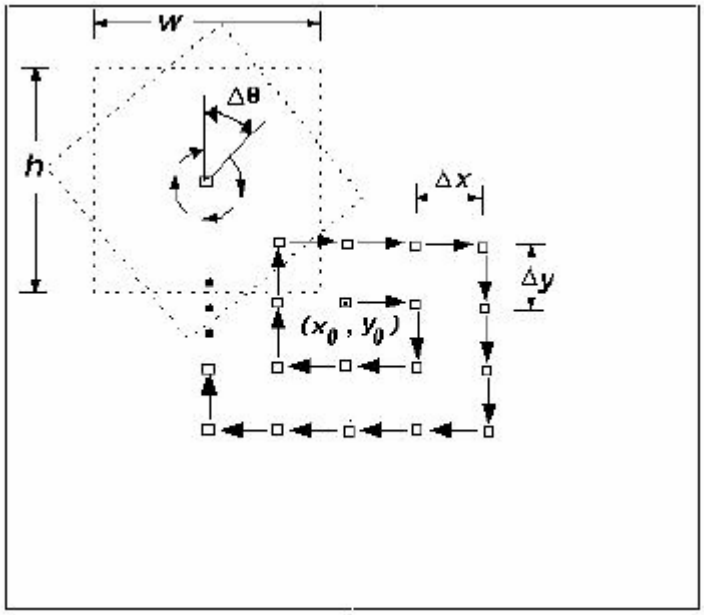

Figure 12: The scheme of the window movement (spiral trajectory).

The image contains three recognized wire terminals (1, 2, 3). The little white square in this figure represents the position where the requested work piece was found and the big square represents the search window attached to this position. At the bottom of the images you can read the position coordinates and orientation found by the system for each recognized work piece. The LIRA classifier is a flexible method, i. e. the system not always recognizes an exact centre of the work piece but the point that is sufficiently near to the work piece.

Sometimes the system can recognize one work piece several times. In Fig. 14 is shown multiple recognition of one work piece (case 1 and 3). In some cases this happens because the system is able to find a work piece far away of its centre point, e. g. marks 2, 3 in Fig 14, it happens because the window movement step ($\Delta x,\ \Delta y,\ \Delta\theta$) is too large, for example mark 1 in Fig 14. Marks 4 (Fig. 14) have sufficient accuracy for a manipulator work.

7. DISCUSSION

We faced with two specific recognition problems. The first problem is to avoid work piece recognition redundancy and the second problem is to improve the accuracy of the work piece finder.

The obtained results are sufficiently good, but the searching method should be improved.

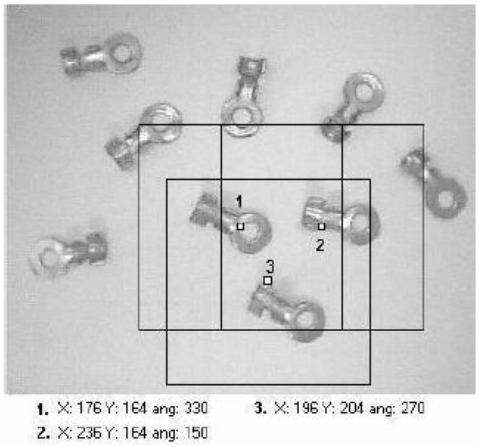

Figure 13: Work pieces position recognition.

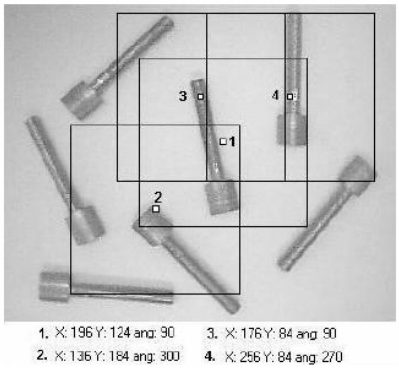

Figure 14: Work pieces position recognition with redundance.

All these experiments were realized with an IntelTM Pentium 4, 2.66 GHz personal computer with 512 MB in RAM.

Microassembly is a sufficiently new area and the range of existing manager software for automatic micro work piece recognition and microassembly is very limited. In order to have our own technological software platform for our research activities we created the OptikRna software. It was created using C++ programming language in concordance

with paradigms of Object Oriented Programming (OOP). C++ was used because it allows to create faster applications in comparison with other programming languages that is important in tasks that demand large computational burden, e. g. neural classifier realization. OOP is perfect to create software for engineering research because it offers great maintenance capabilities and possibility of easy code reuse.

LIRA classifier was used to recognize the micro work pieces of different classes. It demonstrates sufficiently good results in object recognition even if the images are of low quality and have different illuminance conditions. The best results were obtained with normalized images. In the future it is necessary to improve the recognition of the work piece centre, and to add more images to databases.

8. CONCLUSION

The OptikRna software was created to realize LIRA neural classifier and work piece finder. By means this software it is possible to manage the work piece image databases in micro work piece recognition task for micro assembly. It was made in concordance with paradigms of Object Oriented programming and offers many functions combined with friendly Graphical User interface. Software was tested and proved. LIRA neural classifier was adapted for work piece recognition. It was trained and tested for five and eight classes recognition. Every class corresponds to one type of a work pieces, except one that corresponds to the absence of work piece. So the system can recognize the case when there is no work piece in the work area. In the experiments with five classes we used 20 samples

for each class and in the experiments with eight classes we used 40 samples for each class. Images of a tube base, a cone head screw, a round head screw, a cone, a stator axis, an Allen screw and a wire terminal were used to test our system. For five classes, the best recognition rate of 94% was obtained (without distortions) and of 92% with distortions of initial images of training set. For eight classes the recognition rate was 90%. The results are acceptable.

Nevertheless it is necessary to improve the accuracy of work piece position detection. The LIRA classifier has good performance in work piece recognition with the images of low quality. It has also good perspectives for piece position finding.

The training and recognition computer time is rather small. The time for recognition of any sample of any database was less than 0.4 seconds.

9. ACKNOWLEDGEMENT

This work is supported in part by the projects PAPIIT IN116306-3, PAPIIT IN108606-3.

The authors gratefully acknowledge Oleksandr Makeyev for his constructive discussions and helpful comments.

10. REFERENCES

[1] Ofeifer T. & Dussler G., Process observation for the assembly of hybrid microsystems, IEEE International Sysposium on Micromechatronics and Human Science, 2002, pp. 117-123.

[2] Bleuler H., Clavel R., et al., Issues in precision motion control and microhandling, IEEE Procedings of Internationa Conference on Robotics & Automation, 2000, pp. 959-964.

[3] Kussul E., Baidyk T., et al., Development of micromachine tool prototypes for microfactories, Journal of Micromechanical and Microengineer, Vol 12, 2002, pp. 795-813.

[4] Kussul E., Baidyk T., et al., Development of Low Cost Microequipment, International Symposium on Micromechatronics and Human Science, 2002, pp. 125-134, Nagoya, Japan, October.

[5] Friedrich C. & Vasile M., Development of the micromilling process for high- aspect- ratio microstructures, Journal of Microelectromechanical Systems, Vol. 5, 1996, pp. 33-38.

[6] Ooyama N., Kokaji S., et al., Desktop machining microfactory, Proceedings of the 2nd International Worshop on Microfactories, 2000, pp. 14-17, Switzerland, October.

[7] Baidyk T. & Kussul E., Application of neural classifier for flat image recognition in the process of microdevice assembly, Procceding of IEEE International Join Conference on Neural Networks, 2002, Vol 1, pp. 160-164, Hawaii, USA.

[8] Jain R., Schunck B., et al. Machine Vision, McGraw Hill Inc. (1995).

[9] Faugeras O., Three-dimensional Computer Vision, MIT Press, Cambridge, Massachusetts, 1993.

[10] Ehrenmann M., Ambela D, et al., A comparison of four fast vision based object recognition methods for programming by demonstration applications, IEEE International Conference on Robotics & automation, 2000, pp. 1862-1867, San Francisco, CA., USA. [11] Steinhaus P., Prinzipielle Komponenten-Analyse versus multimediale Filterhistogramme, Master Thesis, Universitat Karlsruhe (TH), Germany, July, 1997.

[12] Sun-Ho L., Hyun-Ki H., et al., Assembly part recognition using part-based superquadrics model, IEEE TENCON, 1999, pp. 479-482.

[13] Bennamoun M. & Boashash B., A structural description based vision system for automatic object recognition, IEEE Transactions on Systems, Man and Cybernetics,

Part B, vol. 27, Issue 6, December, 1997, pp. 893-906.

[14] Detter H. & Radjenovic M., Recognition of thin, flat microelectromechanical structures for automation of the assembly process, Journal of intelligent Manufacturing, Vol. 8, No. 3, Jun, 1997, pp. 191-201.

[15] Mitzias D., Mertzios B., A neural multiclassifier system for object recognition in robotic vision applications, Measurement 36, Elsevier, 2004, pp. 315-330.

[16] Mardanov A., Seyfried J. et al., An automated assembly system for a microassembly station, Computer in Industry 38, Elsevier, 1999, pp. 93-102.

[17] Toledo G., Kussul E., et al., Neural classifier LIRA for recognition of micro work pieces and their positions in the processes of microassembly and micromanufacturing, The Seventh All-Ukrainian International Conference on Signal/Image Processing and Pattern Recognition, 2004, pp. 17-20, Kiev, Ukraine, October.

[18] Baidyk T., Kussul E., et al., Flat Image recognition in the process of microdevice assembly, Pattern Recognition Letters, Elsevier, Vol 25, 2004, pp. 107-118.

[19] Rossenblatt F., Principles of Neurodynamics, Spartan books, New York, 1962.

[20] Kussul E. & Baidyk T., Improved method of handwritten digit recognition tested on MNIST database, 15-th International Conference on Vision Interface, 2002, pp.192-197, Calgary, Canada.

AUTHORS BIOGRAPHY

Gengis Kanhg Toledo-Ramírez received the B.S. degree in Electronics Engineering from Technology Institute of Veracruz, Mexico in 1998 and the M.S. degree in Electrical Engineering from the Research and Advanced Studies Center (CINVESTAV) of the Politechnical National Institute, Mexico in 2002. He is currently and undergraduated Engineering Ph. D. student in Mechatronics Engineering at The Applied Sciences and Technology Development Center (CCADET) at UNAM Mexico. His research interests lie at the artificial intelligence, automation processes for microassembly and micromanufacturing and software and space tecnology.

Ernst M. Kussul received his M.S. in mechanics from Leningrad Polytechnic Institute, Russia. He received the Ph.D. in mathematical logic and programming from the Institute of Cybernetics of the Ukrainian Academy of Sciences. In 1967 he joined the Institute of Cybernetics of Ukrainian Academy of Sciences as a junior researcher, from 1976 as a senior researcher, and from 1982 as a leading researcher. In 1982 he received the D. Sci. degree in artificial neural networks from the Institute of Cybernetics of the Ukrainian Academy of Sciences. From 1988 to 1998 he was head of the department of neural networks in the International Research and Training Center of UNESCO/IIP of Information Technologies and Systems. Since 1998 he is Researcher level "C" and the head of the Laboratory of Micromechanics and Mechatronics in UNAM, Mexico. He has published over 140 scientific papers, authored and coauthored 2 books; he is author of 11 patents. He participated in international projects INTAS, ISF. He is laureate of the Prize of the Government of Ukraine in Science and Techniques in 1997. He is a member of the Mexican Academy of Technology. He is a member of IEEE. His present research interests are micromechanics, mechatronics, neural networks, and pattern recognition.

Tatiana N. Baidyk received the M.S. in Electrical Engineering from the Kiev Polytechnic Institute, Ukraine in 1977, and the Ph.D. in control systems for mobile robots in 1983 from the Institute of Cybernetics of the Ukrainian Academy of Sciences. From 1977 to 1995 she was with the Institute of Cybernetics of the Ukrainian Academy of Sciences as a post graduate student from 1977, as a junior researcher from 1980, and as a senior researcher from 1990. In 1994 T.Baidyk received D.Sci. in the area of neural networks from the Institute of Cybernetics of Ukrainian Academy of Sciences. From 1998 she was with the International Research and Training Center of UNESCO/IIP of Information Technologies and Systems as a leading researcher. From 2001 she is Researcher level "A" in UNAM, Mexico. She has published over 120 papers, authored and coauthored 2 books, 3 patents, and edited 8 other books. She is a member of the European Council of Artificial Intelligence, Association of Artificial Intelligence (Russia), a member of the Association of Developers and Users of Intelligent Systems (Ukraine), a member of the Executive Committee of the Neural Network Society (RNNS, Russia). Baidyk´s present research interests are neural networks, pattern recognition, control systems, and industrial applications.