AN OBJECT RECOGNIZING SYSTEM FOR INDUSTRIAL APPLICATIONS

Marcelo Kleber Felisberto
Programa de Pós-Graduação em Engenharia Elétrica e Informática Industrial – CPGEI
Centro Federal de Educação Tecnológica do Paraná – CEFET-PR
Av. Sete de Setembro n° 3165, Curitiba, PR, Brazil
mkf@cpgei.cefetpr.br

Tania Mezzadri Centeno
Programa de Pós-Graduação em Engenharia Elétrica e Informática Industrial – CPGEI
Centro Federal de Educação Tecnológica do Paraná – CEFET-PR
Av. Sete de Setembro n° 3165, Curitiba, PR, Brazil
mezzadri@cpgei.cefetpr.br

Lúcia Valéria Ramos de Arruda
Programa de Pós-Graduação em Engenharia Elétrica e Informática Industrial – CPGEI
Centro Federal de Educação Tecnológica do Paraná – CEFET-PR
Av. Sete de Setembro n° 3165, Curitiba, PR, Brazil
arruda@cpgei.cefetpr.br

Abstract. *On several automatic systems for manufacturing and assembly, specially on Flexible Manufacturing Systems (FMS), identifying mechanical parts produced and detecting their position and orientation are important issues related to the necessity of handling these parts by industrial robots.*
Currently, the recognizing process is performed by human check, but it may lead to increasing errors and accident probabilities. So the implementation of an effective automatic system, in order to recognize the parts, would not only avoid these risks but it would also improve the process velocity as well as its reliability
This work presents an automatic system for free-form objects recognizing which identifies mechanical parts produced by a FMS. This system can recognize objects in a monochromatic image, captured by a charge couple device (CCD) camera. In addition to this, the system can be easily enabled to verify the parts orientation.
This work used the concept of behavior vector, from the image indexing techniques, as a solution for the objects representation. Then, during the recognizing process, at least one hypothesis are generated by a backpropagation neural network trained to recognize the pattern vectors (known objects). Finally, the hypotheses are evaluated through a final verification process.
As a result, the system offers quick and correct answers and also flexibility to be applied in other applications.

Keywords*. object recognition, image indexing, image analysis, backpropagation neural network, FMS.*

1. Introduction

The goal of an object recognition vision system (ORVS) is to find objects on images taken from the real world, using object models which are known a priori (Jain et. all, 1995). Currently, this task is easily performed by humans, but it is surprisingly difficult to unable machines to recognize objects with the same efficiency (Pope, 1994).

Artificial intelligence technologies have leaded to advances on computer vision researches, including new automatic object recognition approaches. However, the object recognition technology is not common on the industry nowadays, even on repetitive tasks (Orth, 1998).

One reason is the limiting performance of some object recognizing systems when applied on real time applications. Another limiting factor is the complexity of the object shapes and the capacity of the system to learn or archive a vast number of patterns keeping the same efficiency and agility (Jain et. all, 1995).

The development and implementation of effective automatic systems, in order to recognize the parts, are an important industrial necessity to diminish human errors and speed up the production (Centeno and Bagatelli, 2000). Besides, the automatic object recognition can support inspection tasks and allow robots and machines to manipulate parts and tools correctly (Rudek et. all, 2001).

This paper makes a briefly review about solution on automatic object recognition approaches and present a new system for object recognition task in order to support industrial manufacturing processes. The system is based on image indexing technique and neural networks.

2. The object recognition problem

A generic model of an automatic ORVS is shown on Fig. (1). The block diagram schemed represents the elementary components of an ORVS and how the information flows through the system during the recognition process. Basically, any automatic object recognition systems are functionally equivalent to this model (Jain et. all, 1995). Generally, the systems differ one from each others by the solution present on each component for the information processing as well as the kind of information that is processed and passed way. The components represented by each block are:

- Model database (or just modelbase)
- Feature detector
- Hypothesizer
- Hypothesis verifier

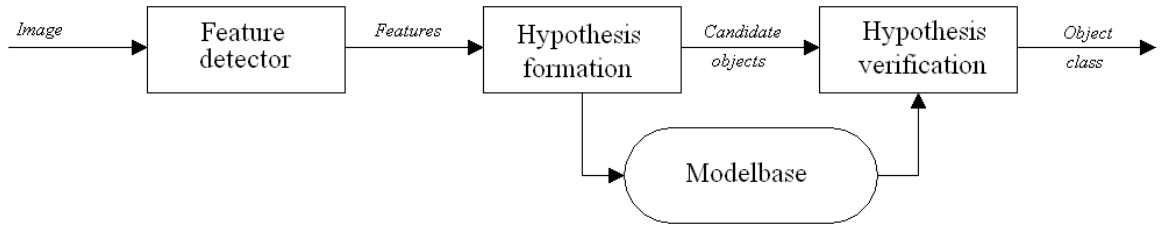

Figure 1: The ORVS block diagram (Jain, 1995)

The model database contains all the models known to the system. The information archived on the model database depends on the approach for the recognition process and the object model representation form. The main idea is to group in a data structure, like a vector or a graph, the features of a specific object model, where each feature can be extracted from an image and describes one or more attributes of the object model (Jain et. all, 1995). Usually, the models are archived using a data structure for indexing/retrieving (i.e., hash tables, lookup tables, trees) to facilitate the access by others components of the system (Jain et. all, 1995).

The feature detector applies operators to images and identifies locations of features that help in forming object hypotheses (Jain et. all, 1995). Generally, the feature detection includes an image segmentation process to identify important regions on the image, corresponding possible objects (Rudek et. all, 2001). After segmenting, properties of these regions, such as area, perimeter, color distribution and others can be evaluated, and relations between these properties can be found. The results usually are grouped on a type of data structure compatible with the object representation form (Jain et. all, 1995).

The both, the hypothesizer and the hypothesis verifier, are responsible for the classification process. That means to recognize the object, based on the group of features extracted from the image by the feature detector. The trivial solution would be to match the group of features with each model of the database and to find the most similar model. Although it is feasible, it is impracticable for real time application because it takes a lot of time. Then, alternatives to select a partition of the model-base, or to select a few model candidates (hypothesis), are very attractive. Actually, the matching procedures would be applied just on a reducing number of database models to detect and verify the better choice (hypothesis verification).

Since it is been used images to identify and recognize objects, the object recognition problem can be treated as an image analysis problem. And it involves three basic stages: distinguishing important regions of the image (image segmentation); measure properties of these regions, or relations among them (properties measurement); and using the values of these properties and relations to classify or describe the entire image or just a specific region of the image (classification) (Rosenfeld, 2001). These stages are closely related to the system components functions. Feature detection, for example, includes image segmentation and properties measurements. And classification is equivalent to the hypothesizer and the hypothesis verifier functions.

2.1 The object representation and the model database

The object representation method can be focused on the object (object centered representation) or on the viewer (viewer centered representation). The first describes an object model by features located in relation to a coordinate system centered on the object. By the other (viewer centered object representation), 2D characteristics views or aspects are used to describe how the object appears from a single viewpoint, or a range of viewpoints yielding similar views (Pope, 1994).

The main advantage of viewer centered representation is to avoid 3D/2D matching procedures during the recognize process, which are much more difficult and time consuming than 2D/2D matching procedures. On the other hand, as complex an object is, as many views are necessary to describe it (Pope, 1994). This problem can be diminished by choosing strategic viewpoints for model description or by using geometric invariant features (Mundy and Zisserman, 1992; Pope, 1994).

The RIO (Relational Indexing of Objects) system (Costa and Shapiro, 2000) use invariants relations between primitives, like straight line segments, arcs and ellipses, or groups of primitives, like coaxial arcs, parallels straight lines segments and line segment joints, to index 3D objects on a lookup table. Each pair of primitives (or groups of primitives) sharing the same arc, line, endpoints or the center point that can be considered as a geometric invariant relation receives an index. Consequently, the models are indexed on a lookup table by the index of its invariant properties. As a result, the RIO system is able to recognize 3D object having planar, cylindrical, and threaded surfaces (Costa and Shapiro, 2000).

Beis and Lowe (1999) developed a method for object recognition where objects are represented by groups of straight-line segments having particular properties. Relations between these straight line segments, like length rations

and angles, are used to construct a feature vector to represent each model. A structure for data indexing and retrieving, known as kd-tree, is used to archive the feature vectors (models) by its value components.

There are a lot of examples of geometric invariants properties that can be used on objects representation and recognition (Mundy and Zisserman, 1992; Abdallah, 2000; Song et all., 2001). But using geometric primitive forms, or invariant relations between them, is a shape-limiting factor. The RIO system (Costa and Shapiro, 2000), for example, is able to recognize just 3D object having planar, cylindrical, and threaded surfaces. And the Beis and Lowe (1999) method is just appropriate to recognize objects that can be sketched by a group of straight-line segments. Besides, colors or gray levels are not considered by these approaches.

A more including solution, which comes from image indexing techniques, was proposed by Rudek (1998). The image of a model is divided in sub-regions and a value, based on the pixels gray level distribution, is calculated to each sub-region. The results are grouped on a vector (called behavior vector) and the sequence of the vector values components is used as an index to access the archived models (or images). This technique is very useful for fast image indexing/retrieving and its use on object representation can offers some important advantages. The behavior vector is color (gray level) sensible (Rudek et. all, 2001) and does not offer restrictions about the objects shapes.

2.2. Feature detection

Image segmentation is a common image processing procedure, and there are many segmentation techniques that can be applied on images for specific regions identification purposes. Some techniques include separating regions by the texture aspects, borders, color, or gray levels (Gonzales, 1987). In some cases it is necessary to use some image pre-processing procedures before segmenting to enhance visual aspects of the image, such as noise reduction, smoothing and background subtraction (Rosenfeld, 2001).

Sometimes, the entities of an image differ in lightness; thus the pixels belonging to image regions that represent different entities have different range of gray levels. Actually, the image can be segmented by comparing the gray levels of the images pixels to some reference value (threshold) and assigning them to classes (darker or lighter pixels) (Rosenfeld, 2001). As a result, a binary (black and white) image is generated. This procedure calls thresholding, and there are a lot of techniques to establish an optimum reference value (threshold) (Sankur and Sezgin, 2001).

After threholding, regions on the image corresponding to possible objects can be easily detected by an image sweeping. Then, many features can be extracted to describe the segmented regions. Area, perimeter, centroid, moments, and relation among some of these features can be used as regions descriptors (Jain et. all, 1995).

Another solution for image segmentation is the edge detection techniques. Pixels that lie on edges can be detected by the fact that there are large differences in the image gray levels in their neighborhoods (Rosenfeld, 2001). Some authors (Beis and Lowe, 1999) (Costa and Shapiro, 2000) use edge detection technique to identify primitives (straight line segments, arcs, ellipses, etc.), and objects are described based on relations between these primitives.

Some useful information can be extracted from the entire image or from some segmented region to describe the image content. Histograms, intensity curves and discrete Fourrier transform are some techniques to extract meaningful information from the image, and it has already been used by Orth (1998) on RAP (Automatic Parts Recognition) system.

The image gray levels histogram is a bar graph in which each bar corresponds to a gray level, and its height indicates the number of pixels having that gray level (Rosenfeld, 2001). Local histograms can be constructed for delimited regions of the image. Rudek (1998) uses local histograms to detect the predominant gray levels range on image sub-regions. The results are grouped on a vector (behavior vector) for image indexing and analysis purposes. This technique has been used on object recognition either (Rudek et. all, 2001; Centeno and Bagatelli, 2000).

2.3. Classification

After feature detection, the group of feature values must be recognized or rejected by the ORVS. Basically, three predominant recognition strategies based on matching, indexing or patterns classification techniques, has been used on object recognition approaches.

Template matching, morphological approaches and analogical methods have been used as matching procedures (Jain et. all, 1995). Basically, the feature values are matched to the models features on the data base and the most similar model are selected. The time dispended on the search usually is a limiting factor (Pope, 1994). It becomes more critical as bigger the model database is. An example is the Hunttenlocher and Rucklidge (1993) approach. They use matching procedures based on the Hausdorff distance for efficiently object recognition, but it is not used on real time applications.

Indexing techniques are alternatives for faster searches. A group of models with similar features is select from the modelbase for the accurate verification. It reduces the space of search to some few hypotheses. Two different indexing techniques are used by Beis and Lowe (1999) and Costa and Shapiro (2000). Beis and Lowe (1999) use a kd-tree structure for feature indexing. The leaves of the kd-tree point to specific groups of models with similar feature values. Costa and Shapiro (2000) uses a lookup table for model indexing based on relational features. Each entry of the lookup table holds a linked list of all models with some particular group of relational features. Usually, on the hypothesis verification stage, matching procedures are used.

Neural networks (Orth, 1998; Abdallah, 2000; Kamgar-Parsi et. all, 2001), nearest neighbor classifiers and bayesian classifiers are some patterns classification techniques that have been used on object recognition tasks (Jain et. all, 1995). On these approaches, each model represents a class. And the features values of the extracted features are used to classify them. The neural networks are especially attractive for many cases as a result of their ability to partition the feature space using non-linear boundaries of classes (Jain et. all, 1995).

3. Image acquisition and background conditions

To recognize objects from single intensity images is a complex image analysis problem (Abdallah, 2000). In fact, the complexity of the object recognition problem depends on several factors (Jain et. all, 1995):

1. Scene aspects: illumination, background, camera parameters and viewpoint.
2. Number of models which must to be known a priori.
3. Number of objects on the same scene and possibility of occlusion.
4. The geometric complexity of the objects.

In order to provide appropriate conditions for image acquisition, an inspection cabin, as shown on Fig. (2a), was constructed. The cabin can be located on strategic positions of the FMS. It was equipped with a video camera charge couple device (CCD camera) and a backlight system. The camera is a WAT-202-B CCD WATEC model, and is connected to a PC Pentium III 750Mhz station by a digitizer board. When a mechanical part passes through the cabin, the image is captured by the CCD camera, digitized by the digitizer board and processed by the recognizer system. The camera and the recognizer system can be started by signals that come from position sensors of the FMS.

Usually, on a FMS and others automatic manufacturing systems, the parts are supported by pallets and moved by conveyers. Actually, the pallet restricts the part movements. As shown on Fig. (2b), the part rotation is only possible through the Z axis.

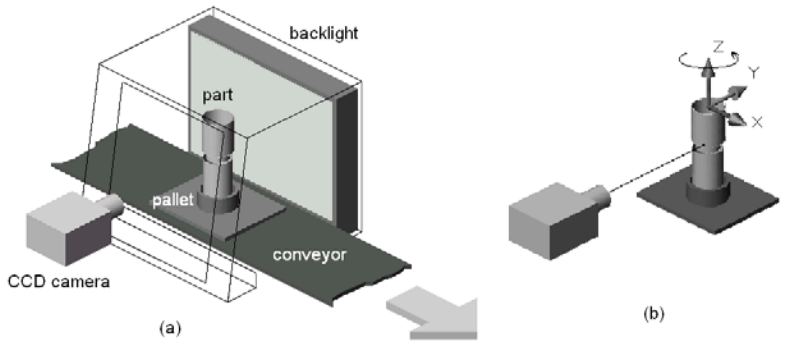

Figure 2: (a) The inspection cabin for image acquisition. (b) A coordinate system fixed on the part.

It is used monochromatic images with 256 gray levels, where 0 means black and 255 means white. The others (1-254) are the intermediary gray levels. These images are two-dimensional and do not provide depth information.

Just one pallet supporting one object passes through the inspection cabin at time. Actually, there are no possibilities to inclusions of one object by others. The objects are considered rigid, opaque and not articulated bodies. On this work there were no pretensions to develop an object recognizing system to articulated, amorphous or transparent objects recognition.

The industrial applications, in which the object recognition system is intended to perform, are considered real time applications. Therefore, the recognition processing time can not dispend much more then some few seconds. The manufacturing process can not be delayed or interrupted by the recognition process.

4. Methodology

The system works on two stages: training and recognizing. The both involve image processing, indexing and neural network techniques. The image pre-processing, segmentation and feature extraction are identical. However on the following steps, the stages differ. On the training stage a neural network is trained and the patterns are archived. And on the recognizing stage, there are the hypothesis formulation and verification tasks.

4.1. Image pre-processing and segmentation

The illumination system of the inspection cabin (backlight) provides images with a high contrast between the object and the background, witch can be considered static. On these conditions, the background subtraction from the image can be done by algebraic image operators. An image with no objects, take from the same angle, is used as the background image. By adding the object image to the inversed background, part of the exceeded information of the image is eliminated (Fig. (3b)).

Variation in lightness can produce some noise and shadow. A thresholding operation is used to diminish these effects. The Otsu method (Otsu, 1979) is used to find the best reference value to the image thresholding. After

thresholding, the image is added to the original image to recover some details of the object that was lost during the thresholding process. The sequence of the results is shown on Fig. (3).

(a) (b) (c)

Figure 3: (a) Original image; (b) background subtraction; (c) thresholding and addition with the original image.

4.2. Feature extraction

After the image segmentation, the rectangle, in which the object is enclosed (called including rectangle), is found by an image sweeping. The background image can be used to determine the inferior limit of the rectangle. A value called dimensional factor (Fd), based on the vertical and horizontal dimensions of the rectangle, is calculated by the Eq. (1).

$$F_d = \frac{h}{v+h} \tag{1}$$

The values 'h' and 'v' are the horizontal and the vertical dimensions of the rectangle. Consequently, the Fd values are smaller as thinner objects are, and it is always a value between 0 and 1, since 'h' and 'v' are never less than 0. A scale can be used to classify the Fd value. Using a scale with five Fd classes (number of classes $N_C = 5$), for example, the Fd classification would follow the Tab. (1), where each Fd class correspond to a range of Fd values. The number of classes N_C is a system parameter that can be modified by the user.

Table 1 – The Fd classes and their respective Fd range values

Fd class	Fd range
1	0,0 ≤ Fd < 0,2
2	0,2 ≤ Fd < 0,4
3	0,4 ≤ Fd < 0,6
4	0,6 ≤ Fd < 0,8
5	0,8 ≤ Fd ≤ 1,0

The 5 following steps describe the behavior vector construction to the part of the image delimited by the including rectangle:

1 Divide the rectangle in n×m blocks ('n' lines and 'm' columns)

2 For each $block_{ij}$ (where i and j correspond the respective line and column of the block), do the steps 3 until 5, following the sequence: $block_{11}$, $block_{12}$, …, $block_{1m}$, $block_{21}$, $block_{22}$, … $block_{2m}$, …, $block_{nm}$.

3 Count the number of pixels belonging to each range of gray levels. The Tab. (2) shows the range value for each range of gray levels, considering 16 gray levels ranges (number of gray level ranges: Nr = 16).

4 Determine the predominant range of gray levels on the block and its respective range value (in case of equality, takes the maximum range value)

5 Assign to the current block the value of the predominant range of gray levels

The block values will correspond to the behavior vector elements on the same sequence the values were calculated.

Table 2: The values for 16 gray levels ranges – from (Rudek,1998)

range value	range of gray levels	range value	range of gray levels
00	-	08	119-135
01	000-016	09	136-152
02	017-033	A	153-169
03	034-050	B	170-186
04	051-067	C	187-203
05	068-084	D	204-220
06	085-101	E	221-237
07	102-118	F	238-255

The Fig. (4) illustrates the behavior vector construction for an image. The behavior vector and the Fd class are the results of the feature extraction process.

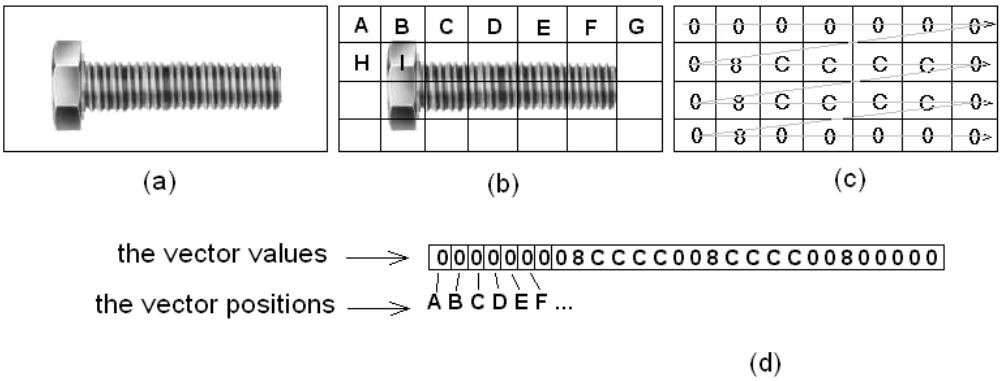

Figure 4: (a) The pre-processed image; (b) the sub-regions/blocks; (c) the blocks values; (d) the behavior vector.

4.3. Object representation

To represent the objects by only one vector, another technique was used to modify the behavior vector using the Fd class value with no lost of dates. This technique includes two steps:

1 A vector with (m × (Nc - 1)) null positions is aggregated to the behavior vector at the end of it.

2 The behavior vector values are flipped, to the right, through a (m × (Fd class - 1)) numbers of null positions.

As a result, the behavior vector are expanded from "m × n" to "m × (n + Nc - 1)" and the number of null positions at the beginning or at the end of the vector is a function of the object Fd class.

The models on the database are vectors generated through the same procedures.

4.4. The neural network

A backpropagation neural network (BPN) was implemented to support the recognition task. This kind of neural network always has an input, an output and one or more intermediated (hidden) layers of processing elements (PEs) (Murshed, 1995). The Fig. (5) shows a typical BPN with only one hidden layer. The outputs from the PEs of the input layer, after multiplying with the corresponding interconnecting weights, serve as inputs to the PEs of the hidden layer. The outputs from the PEs of the hidden layer, after multiplying with the corresponding interconnecting weights, serve as inputs to the PEs of the output layer. A bias processing element supplies a constant output of +1 to all the PEs of the hidden and the output layers. BPN models with more than one hidden layer process information on the same principle (Zafar and Mohamad, 2002). Appropriated connection weights values are found by the BPN training.

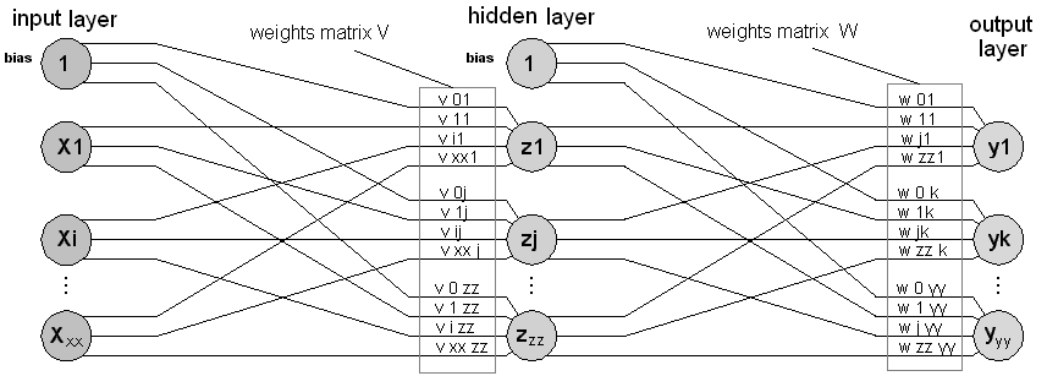

Figure 5: A backpropagation neural network with 1 hidden layer.

For a PE, the output is typically a function of the sum of input into it. As shown on the Fig. (6), the sum of PE inputs is the argument of a transference function, witch produces the PE output. Typically, the transference function for BPN models is linear, sigmoid or hyperbolic tangent (Fausset, 1994). Using sigmoid transference function on the hidden and output PEs, the network output becomes a high non-linear function of the network input (Zafar and Mohamad, 2002). As a consequence, for each output PE, the network can be thought as representing a non-linear function of the inputs. For "xx" PEs in the input layer and "yy" in the output layer, the network represents "yy" non-linear functions of "xx" variables (Zafar and Mohamad, 2002).

One way to transform the vectors on adequate network input signal is to binarize their values (Skapura,1996). The chosen rule used to binarize the vectors just replace each vector element by a number with "Nr-1" bits (Nr = number of gray level ranges). For example, when Nr = 4, the range values 0, 1, 2 and 3, are replaced respectively by 000, 001, 011 and 111. Actually, the vector {1 0 0 2 2 3 3 0} would become {001 000 000 011 011 111 111 000}.

Before the network training, each binarized vector is matched to the others, and the same positions with the same value are labeled for all. These vector positions are disabled and they are not considered by the neural network in any stage. But these positions are not completely discarded. They come back to be considered on the hypothesis verification process.

This strategy provides two main advantages:

1 Just the essential information to differentiate the patterns is considered.

2 The neural network works with a reduced amount of data, accelerating the training.

After all, the reduced vectors are ready to be used by the neural network. The next sub-topic will describe the backpropagation training algorithm.

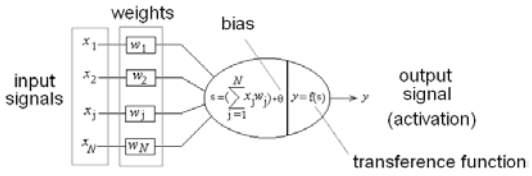

Figure 6: A processing element.

5.5. The backpropagation training

The BPN training is supervised, i.e., training is based on a group of known vector pairs (the input and the target vectors). The input vectors are the reduced vectors for the patterns (models). And the target vectors are binaries representations of the pattern labels.

The following steps resume the backpropagation training (Fausset, 1994):

0. Initialize connection weights (Set to small random values)
1. While stopping condition is false, do the steps 2-7
2. For each pair (input vector, target vector) do the steps 3-6.
3. Load the input vector on the input layer and propagate the signals through the network PEs until the output layer.
4. Match the outputs to the target vector elements and calculate the square error.
5. As the error is backpropagated through the network, calculate the weights updates.
6. Update the weights and bias values.
7. Test the stop conditions (error < maximum error).

4.6. Implementing a BPN for the hypothesis formulation

It was implemented a BPN with only one hidden layer. Some problems can be easier solved using two or more layers, but it had been proved that if a problem can be solved by a BPN with two or more hidden layers, the same problem can be solved by a BPN with just one hidden layer (Fausset, 1994).

The input layer just passes way the input values with no processing. So the number of PEs on the input layer is defined by the input vector size (the reduced vector). Each output layer represents a pattern. Consequently, the number of PEs on the output layer is defined by the numbers of patterns.

The transference function of the hidden and the output PEs is the bipolar sigmoid (equation 2). The bipolar sigmoid behavior is almost identical to the simple sigmoid, but its output varies from -1 to +1 as on the hyperbolic tangent (Fausset, 1994).

$$f_{sb}(x) = \frac{2}{1+e^{-x}} - 1 \qquad (2)$$

There are no specific rules to determine how many PE's must be on the hidden layer (Murshed, 1995). For convenience, the number of PE's on the hidden layer (zz) was define on these work as a function of the input vector size (xx), using a linear relation: $zz = c \times xx$, where "c" is a constant. After exhaustive series of tests, it was defined $c = 1/2$.

When a new input vector is presented to the network, it produces an output. If just one of the output PEs are activated (activation value more than +0.75 for the activated PE, and less than -0.75 for the others) it means that the network classified the input vector as one of the patterns, and just one hypothesis is formulated by the system. But if it does not happen, 3 hypotheses are formulated, based on the 3 most activated PEs.

Since some positions of the original vector are not processed by the network, it is necessary to verify all the hypotheses, even if there is just one.

4.7 Hypothesis verification

On the verification process, the complete image vector (not the reduced) is matched to each hypothesis models. The sum of the absolute errors is calculated. The hypothesis with the minor error is selected. If the hypothesis error value is

acceptable, the hypothesis is validated. On the contrary, all hypotheses are rejected. The maximum acceptable error percentage (e) is a system parameter that can be modified by the user.

5. The implementation

The system was implemented on two modules: training and recognizing. It was used C++ object oriented programming language on windows platform. Basically, the recognizing module is enabled, in order to recognize a specific group of patterns, by the training module.

There are some parameters used by the training module that can be modified by the user. They are:

- the working directory
- the models images
- the background image
- the patterns labels
- the n × m dimensions for the behavior vector
- N_{cl} (number of classes for the including rectangle F_d)
- N_r (number of ranges for gray levels values)
- e (the maximum error percentage for the hypothesis verification)

The recognizing module is automatic and there are no parameters to be set by the user, besides the input image and the working directory. Many parameters are calculated or transferred to the recognizing module database during the training. Various recognizing modules can be trained by the same training module in order to recognize different groups of objects. On the tests, 3 recognizing module was trained based on very different kind of objects groups.

6. Tests and results

For the test images acquisition was used a prototype of the inspection cabin constructed on the digital image processing laboratory (PDI-NEHOS) on CEFET-PR. It was used a CCD WATEC camera (model WAT-202-B) connected to a PC Pentium III 750Mhz station.

After training, the possible responses on the recognizing module to an input image are:

- a pattern label
- no detection

When the input image is a known object, the system response is:

- right, when the object is recognized as the correct pattern (right pattern)
- wrong, when the object is recognized as another pattern (wrong pattern)
- wrong, when the object is not recognized as any pattern (no detection)

When the input image is an unknown object, the system response is:

- right, when the object is not recognized as any pattern (no detection)
- wrong, when the object is recognized as a pattern (wrong pattern)

On the first test (chess pieces test) it was used 7 patterns, as shown on the Fig. (7). For the horse piece it was used 2 images covering different viewpoints. A BPN having 241/120/7 PEs on the respective input/hidden/output layers was constructed and trained. The total training time was close to 3 minutes. After training, the recognizing module (called chess pieces recognizer) was submitted to 2 test series:

- One includes 29 chess pieces images (no image test was used on training) that the system must to recognize.
- The other includes 8 images from other different objects that the system must to reject.

As a result, the chess pieces recognizer performance was 100% for the both tests. The Fig. (8) shows the chess pieces recognizer response for one of the images used on the first test series.

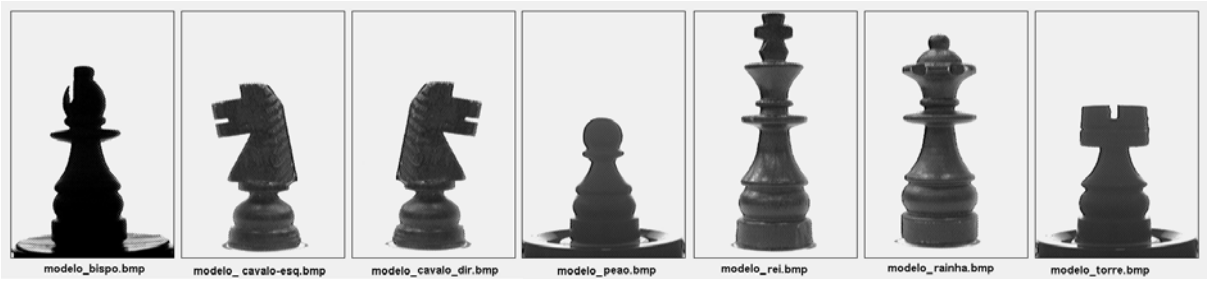

Figure 7: Chess pieces images used by the training module for the chess pieces test.

On the second test (faucet parts test) it was used 12 patterns, as shown on the Fig. (9). For some objects it was used more than one model image covering different ranges of viewpoints. A BPN having 275/137/12 PEs on the respective input/hidden/output layers was constructed and trained. The total training time was close to 36.5 minutes. After training, the recognizing module (called faucet parts recognizer) was submitted to 2 test series:

- One includes 74 faucet parts images (no image test was used on training) that the system must to recognize.

- The other includes 20 images from other different objects that the system must to reject.

As a result, the chess pieces recognizer performance was 93% for the first series (there were 5 no detection) and 100% for the second test.

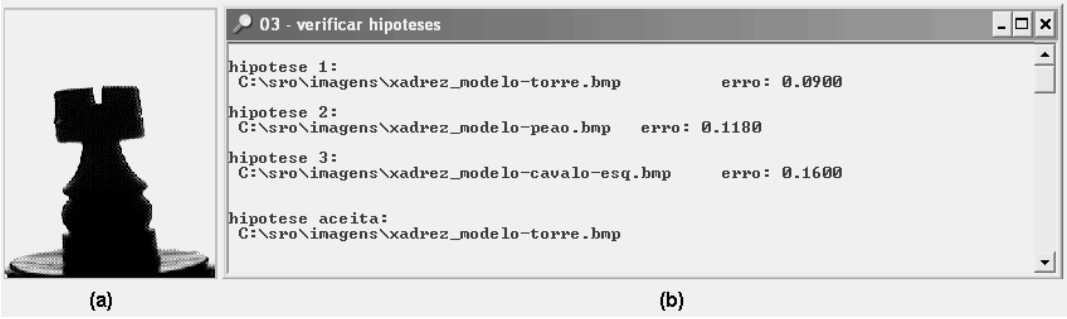

Figure 8: The chess pieces recognizer response (b) to the input image (a).

Figure (9): Faucet parts images used by the training module for the faucet parts test.

The third test (cars models test) intends to test the system in order to recognize more complex objects, giving the object orientation either. It was used 16 patterns (8 viewpoints for each object), as shown on the Fig. (10). A BPN having 255/127/16 PEs on the respective input/hidden/output layers was constructed and trained. The total training time was close to 1h45'. After training, the recognizing module (called cars models recognizer) was submitted to 2 test series:

- One includes 19 cars models images (no image test was used on training) that the system must to recognize.
- The other includes 20 images from other different car models that the system must to reject.

As a result, the chess pieces recognizer performance was 89,5% for the first series (there were 2 no detection) and 81,8% (there were 4 wrong responses) for the second test. The recognizing module response was very fast (i.e. fractions of second) and it was not affected by the training time.

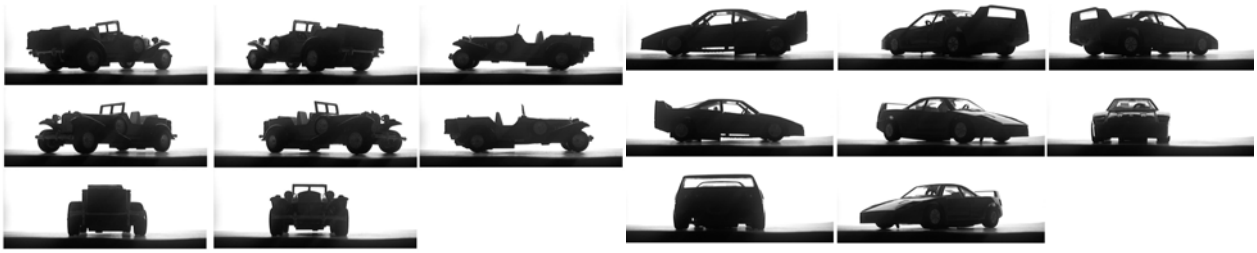

Figure (10): Small cars images used by the training module for the faucet parts test.

7. Future works

Although there was no pretentious to develop a system in order to recognize partial occluded objects, a series of tests with some images, having certain objects regions occluded, revels some surprising results. Actually, the image (a) of the Fig. (11) was modified to give rise others images ((b) until (h)), whose object appears partially occluded.

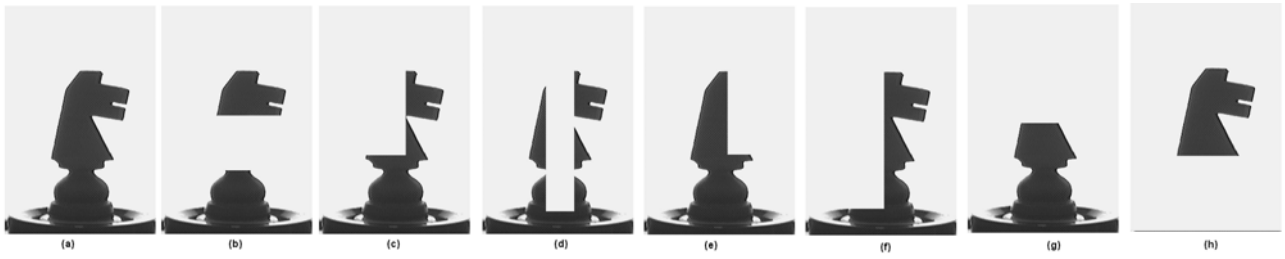

Figure (11): (a) Original image; (b) (c) (d) (e) (f) (g) and (h): Modified images simulating partial occlusion.

The chess pieces recognizer was used to recognize the modified images. It was able to recognize correctly 5 images: (b) (c) (d) (e) and (h). The image (f) was rejected, and the image (g) was recognized as the pattern "modelo_torre". These results have encouraged the researches in order to ability the system for the partial occluded objects recognition on more complex scenes.

8. Conclusions

A new method for automatic object recognition was developed, implemented and tested. On the test, the system performance varied as a function of the object complexity, but the results remained goods (from 85.5 to 100%). Other systems (Orth, 1998; Abdallah, 2000; Costa and Shapiro, 2000) have presented almost the same performance on test, but using objects with simpler shapes. The recognizing modules responses were very fast (fractions of seconds) and it enables the system for many applications involving real time. By using various recognizing modules it is possible to increase substantially the number of patterns that can be recognized. The system performance on tests justifies its implementation for many industrial applications. At the moment, new techniques are been tested in order to reduce the training process time and to increase the system performance for the most critical situations (cars models test, for example).

9. Acknowledgment

This work has been partially supported by Agência Nacional do Petróleo (ANP), Financiadora de Estudos e Projetos (FINEP) - ANP/MCT (PRH10-CEFET-PR) and CAPES.

10. References

Abdallah, S. M. "Object Recognition via Invariance", PhD Thesis, The University of Sydney, Sydney, 2000

Beis, J.S., Lowe, D.G.. "Indexing without Invariants in 3D Object Recognition", IEEE Transactions on Pattern Analysis and Machine Intelligence, vol.21, n. 10, October 1999

Centeno, T. M., Bagatelli, R. C. "Indexing Techniques applied in Machine Vision for Object Recognition", Irish Machine Vision & Image Processing Conference. Belfast: ago. 2000

Costa, M. S., Shapiro, L. G. "3D Object Recognition and Pose with Relational Indexing", Computer Vision and Image Understanding. Seattle, Academic Press, n. 79, p.364-407, 2000

Fausett, L. "Fundamentals of Neural Networks: Architectures, Algorithms, and Applications". Upper Saddle River: Prentice-Hall, 1994. 461 p.

Gonzalez, R.C., Wintz, P. "Digital Image Processing" 2nd. edition, Addison-Wesley, 1987

Hunttenlocher, D., Rucklidge, W. "A multi-resolution technique for comparing images using the Hausdorff distance", IEEE Conference on CVPR, 1993

Jain, R., Kasturi, R., Schunck, B. G. "Machine Vision", New York: McGraw-Hill and MIT Press, 1995

Kamgar-Parsi, Behrooz, Kamgar-Parsi, Behzad, Jain, A.K., Dayhoff, J.E. "Aircraft Detection: A Case Study in Using Human Similarity Measure", IEEE Transactions on Patter Analysis and Machine Intelligence, vol.23, n. 12, December 2001

Mundy, J., Zisserman, A. "Geometric Invariance in Computer Vision". Cambridge: MIT: Cambridge, 1992

Murshed, Nabeel A. "Abordagem Natural Baseada em Redes Neurais para Verificação de Assinaturas", Dissertação de Mestrado, CPGEI CEFET-PR. Curitiba: 1995

Orth, A. "Desenvolvimento e implementação de um Sistema de Reconhecimento Automático de Peças Mecânicas em uma Célula Flexível de Manufatura" – Projeto RAP Laboratório de Automação Industrial, UFSC. Congresso da Sociedade Brasileira de Computação, SBC 1998.

Otsu, N. "A Threshold Selection Method from Gray-Level Histogram", IEEE Trans. on Systems, Man, and Cybernetics SMC-9, pp. 62-66, Jan. 1979.

Pope, A. R "Model Based Object Recognition" – A Survey of Recent Tecniques, Technical Report, jan. 1994

Rosenfeld, A. "From Image Analysis to Computer Vision: An Annotated Bibliography, 1955-1979", Computer Vision and Image Understanding. Seattle, Academic Press, n. 84, p.298-324, 2001

Rudek, M. "Uma Proposta para Indexação e Recuperação Automática de Imagens e Reconhecimento de Cheques Bancários Baseadas no Vetor de Comportamento", Dissertação de Mestrado, CPGEI CEFET-PR. Curitiba: 1998

Rudek, M., Coelho, L. S., Canciglieri JR., O., "Visão Computacional Aplicada a Sistemas Produtivos: Fundamentos e Estudo de Caso", XXI Encontro Nacional de Engenharia de Produção - 2001, Salvador: 2001

Sankur, B., Sezgin, M. "Image Thresholding Techniques: A Survey Over Categories", (under review) Pattern Recognition, 2001.

Skapura, D. M. "Building Neural Networks" New York: ACM, 1996

Song, B.S., Lee, K.M., Lee, S.U.. "Model-Based Object Recognition Using Geometric Invariants of Points and Lines", Computer Vision and Image Understanding. Seattle, Academic Press, n. 84, p.361-383, 2001

Zafar, M. F., Mohamad, D. "Size, Scale and Rotation Invariant Proposed Feature vectors for Trademark Recognition", International Technical Conference On Circuits/Systems, Computers and Communications, Phuket, Thailand, 2002.