Some Features of Classifiers Implementation for Object Recognition in Specialized Computer systems

Amer Tahseen Abu-Jassar [1], Yasser Mohammad Al-Sharo [1], Vyacheslav Lyashenko [2], Svitlana Sotnik [3]

[1]*Faculty of Computer Science and Information Technology, Ajloun National University, Jordan*
[2]*Department of Media Systems and Technology, Kharkiv National University of RadioElectronics, Ukraine*
[3]*Department of Computer-Integrated Technologies, Automation and Mechatronics, Kharkiv National University of RadioElectronics, Ukraine*

***Abstract* – Generalized formalization of recognition algorithm for specialized computer systems is presented in this paper. The structure features of image recognition methods, which have to be taken into account when developing classifiers for object recognition in specialized computer systems, are described. The fundamental types of images characteristics-features, which are used in various methods of image recognition, are discussed. Approaches to development of classifiers for recognizing robotization objects, which are implemented on basis of Haar classifiers, are discussed. The issues of using machine learning algorithm of adaptive gain AdaBoost for development of such classifiers are also considered. Utilities have been developed for implementation of classifiers for object recognition in specialized computer systems.**

***Keywords* – specialized computer systems, robotics, semantic network, recognition, image, classifier.**

DOI: 10.18421/TEM104-21
https://doi.org/10.18421/TEM104-21

Corresponding author: Vyacheslav Lyashenko,
Kharkiv National University of RadioElectronics, Ukraine
Email: viacheslav.liashenko@nure.ua

Received: 30 July 2021.
Revised: 08 October 2021.
Accepted: 14 October 2021.
Published: 26 November 2021.

1. Introduction

The constant growth of robots' range and expansion of tasks range to the one that can be automated, leads to the need to improve efficiency of a number of processes based on the use of various models and specialized computer systems.

For example, at present, semantic model of knowledge representation is successfully used to improve efficiency of recognition processes and information processing. This is an information model of subject area, which has form of a directed graph, vertices which correspond to objects of subject area, and arcs (edges) that define relationship between them [1].

As objects of subject area, there can be concepts, events, properties, processes. The key idea of such modeling is that model represents data about real objects and connections between them in direct way, which greatly facilitates access to knowledge: by starting from certain concept, and other concepts can be reached along arcs of relations.

In course of semantic modeling, there is requirement – associativity, that is, grouping of information around facts, attributes and objects [1]. At the same time, advantage of using Semantic Networks (SN) is that knowledge that is presented with help of SN lends itself well to processing in specialized computer systems (SCS), for example, robotic vision systems. This is ensured by explicitly specifying links between objects, and it allows us to decipher meaning of original image given by semantic network. SN can also be means of describing relationships between any types of objects, for example, detected and identified in image [2]. Then, in the process of recognition and identification of information obtained in technical vision systems (TVS), it is possible to determine geometric characteristics of objects and their belonging to certain class. The data obtained allows one to identify semantic links between individual objects and correspondence of detected links to standard descriptions (templates).

Thus, it is proposed to develop classifier for object recognition in SCS using example of robotics, which can be further used to organize interaction in "human-robot" system and explore possibility of using SN model to determine meaningful and semantic relationships between objects in robot's workspace.

2. Materials and Methods

2.1. Related Work

A sufficient number of works are devoted to development of classifiers, semantic networks, identification and recognition of objects, number of which is growing every day.

There are many works in which authors believe that today one should not use Haar cascades to detect anything. In such works they say that a lot of neural networks have appeared that work is faster and better. The phones now have hardware support for execution of neural networks, comprising a lot of simple and convenient frameworks for training and executing neural networks [3]. Nevertheless, Haar cascades remain an object of research. For example, an improved Haar cascade classifier that effectively detects and recognizes objects is presented in [4]. The positive image is factual confident image whereas negative image is everything else other than confident image.

In [5], authors propose a method based on choice of classical and deep learning functions. The proposed method, like many others, is based on CNN trainable model, and object classification process includes three stages: data replenishment is performed at first stage of creating a balance database; deep learning features extracted from a pretrained CNN model called Inception V3; using new Joint Entropy method together with KNN (JEKNN) to select best features.

With regard to object recognition, directly in field of robotics, then, pattern recognition and image processing is presented in [6]. Here, authors focused on three different types of objects – three types of hexagons that are classified using deep learning algorithms based on convolutional neural network architectures.

The work deals with a robotic arm, which can be controlled via Telegram application, and also provides ability to work from an Android or IOS cell phone. The method proposed in [6] includes four stages: the first is design, implementation, and control of robotic arm; the second is capture, classification and processing of images; the third allows nut to be clamped through back of robot. Kinematic; the fourth step is changeover of hex nut to appropriate container.

The deep learning method, which is little taken into account in robotic applications, where an algorithm based on faster R-CNN and CNN regression is disclosed is described by authors in [7]. The work uses convolutional networks, namely, two convolutional architectures: for classifying and locating three types of objects; to determine gripping angle for robotic gripping.

There are several CNN-based deep learning methods that can detect objects in an image, among them DAG-CNN, R-CNN, fast R-CNN, Yolo and faster R-CNN [8], [9], [10].

In [8], neural network, structured as DAG (Directed Acyclic Graph), allows us to classify objects found on table by robotic arm. The authors describe process of modeling an extended sorting process by two robotic arms. The paper describes process of highlighting features of recognition objects on different background, changing lighting, noise and shadow.

The features of R-CNN application are described in [9]. The idea is to isolate a specific unit (function) in network and use it as if it were an object detector. The work performs area classification – standard semantic segmentation method that allows us to easily apply R-CNN to subsets with lowest performance within six different object characteristics (overlap, truncation, bounding box area, aspect ratio, viewpoint, part visibility). That is, authors combine two key ideas: 1) ability to apply high-bandwidth convolutional neural networks (CNNs) to ascending region sentences to localize and segment objects; 2) if data are scarce during training, then subsequent fine-tuning for a specific area is implemented, which gives significant increase in performance.

Fast R-CNN and Region Proposal Network (RPN) are described in [10]. RPN is viewed as fully convolutional network that simultaneously predicts object boundaries and objectness scores at each position.

RPN takes an image (of any size) as input and outputs a set of rectangular objects suggestions, each with an objectivity score. Both RPN and Fast R-CNN trained independently will change their convolutional layers differently. As a result, technique is proposed that allows sharing of convolutional layers between two networks.

In [11], we are also talking about Fast R-CNN. The basic idea is that RPN is trained end-to-end to generate high quality region proposals that Fast R-CNN uses for discovery.

In [12], YOLO is presented – approach to object detection. YOLO model also processes images in real time. Only very general representations of objects can be represented with YOLO. This deep learning method outperforms other detection methods, including DPM and R-CNN, in generalizing natural images to other areas such as artwork, that is, YOLO works on both art and natural images from Internet. Fast YOLO, handles an astounding 155fps.

In [13], we are talking about assistive robotics and possibility of using three convolutional neural networks that are trained, one for recognizing group of tools, second for speech recognition, trained with names of tools, and third for recognition of user's hand, taking in consideration classification of 2 gestures: open and closed hand. As a result, authors have developed functional application for assistive robotics capable of receiving and correctly identifying voice commands, associating them with physical object, picking it up and delivering it to user's hands.

2.2. *Structure Features of Image Recognition Methods*

With all variety of different algorithms and methods for image recognition, traditional approach to recognition consists of three components, which is shown in Figure 1 [14], [15], [16].

Any recognition algorithm can be represented as an abstract functional system R, consisting of three components [17], [18], [19]:

$$R = \{A, S, P\},$$

where $A = \{A_k\}$, k = 1, … ,K – alphabet of classes – set of categories in which you need to distribute images;

$S = \{S_j\}$, j = 1, ... ,J – dictionary of features – set of characteristics from which description of an image is made;

$P = \{P_l\}$, l = 1, ... ,L – many decision-making rules.

Components A, S represent informational part of system, and P – methodological one.

The meaning of class concept for different ways of describing images will be different. In turn, way of describing image depends on physical nature of recognition objects and possibilities of formalizing concepts corresponding to them. Decision-making methods are naturally interconnected with way of recognition objects representing.

Thus, information received from TVS goes through following stages:

- transformation of original image into initial representation (it may include both preprocessing and mathematical transformations, for example, calculation of principal components) [14];
- selection of key characteristics (for example, first principal components or coefficients of discrete cosine transform are taken);
- classification (modeling) mechanism: cluster model, metric, neural network, etc.

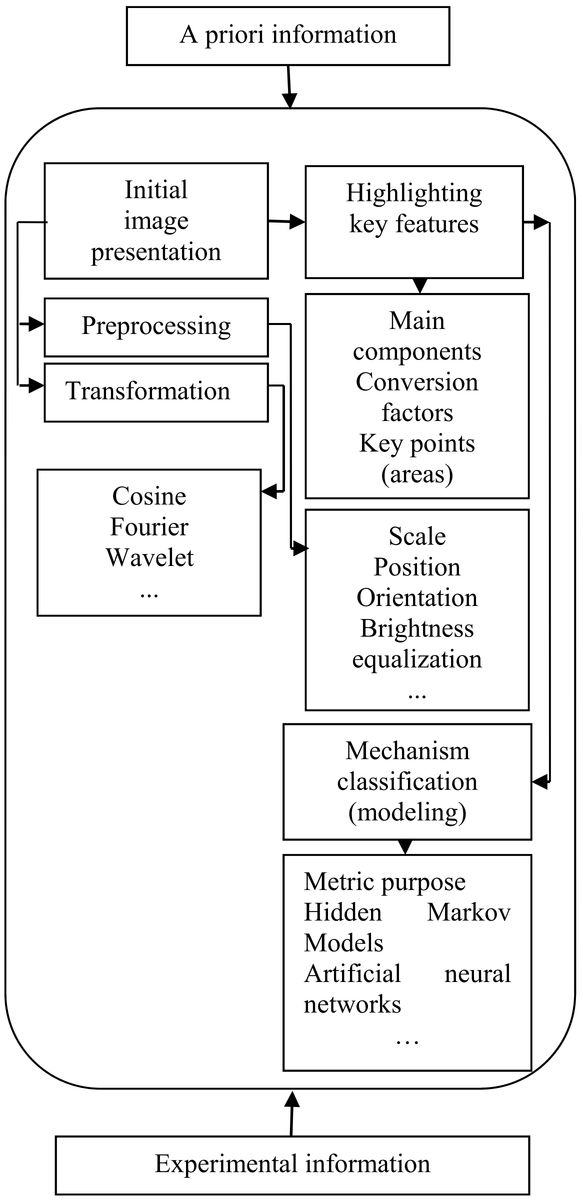

A priori information
Initial image presentation
Highlighting key features
Preprocessing
Transformation
Main components Conversion factors Key points (areas)
Cosine Fourier Wavelet ...
Scale Position Orientation Brightness equalization ...
Mechanism classification (modeling)
Metric purpose Hidden Markov Models Artificial neural networks …
Experimental information

Figure 1. Structure of image recognition methods

In addition, construction of recognition method is based on a priori information about subject area (in this case, characteristics of robotization objects), and is corrected by experimental information that appears during development of method.

Allocation of key features can be based on:

- method of principal components;
- conversion factors;
- highlighting key points (areas).

Principal Component Analysis (PCA) is used to compress information without significant loss of information content. It consists of linear orthogonal transformation of an input vector X of dimension N into an output vector Y of dimension M, where N> M [20].

It should be noted that there are three main types of characteristics-signs [21], [22]:

1. Physical characteristics, such as readings taken from various sensors. Physical characteristics can be deterministic and probabilistic. It is described using vectors.
2. Qualitative characteristics. Examples of qualitative characteristics are concepts "dark", "light", "high", etc. Such characteristics can be described using so-called linguistic variables.
3. Structural characteristics. These characteristics are used to describe images of complex objects or scenes. When describing structural characteristics, certain formal language is used (for example, graph theory, see Figure 2) [22].
4. Logical characteristics are statements about which it makes sense to speak true or false.

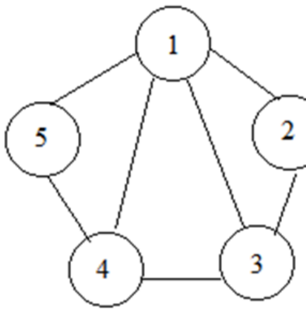


Figure 2. Example of structural characteristics description

The rules for correlating an image to one of classes are called a classifier and are implemented in classification block. The method of assigning an element to any image is called decision rule. An important concept is metrics – way to determine distance between elements of universal set.

The smaller this distance, more similar objects (symbols, sounds, etc.) are, that is, what is recognized. Typically, items are specified as set of numbers, and metric as function. The efficiency of program depends on choice of image representation and implementation of metric; one recognition algorithm with different metrics will make mistakes with different frequency [23], [24].

The most common measure of similarity is distance between points-images in space of measurements (features) X. In most cases, Euclidean metric is used [25].

$$||a-b||=\sqrt{\sum_{i=1}^{n}(a_j^2-b_j^2)}.$$

In some cases, other similarity measures can be used, other than distance measure, for example, cos α, where α is angle between vectors a and b (a measure of correlation).

In many cases, instead of Euclidean metric, simply squares of distances are used, which, without affecting result, greatly simplifies calculations.

3. Development of Classifiers for Object Recognition

When choosing approaches to developing a classifier for object recognition in specialized computer systems, we will consider a semantic model of robot's workspace.

So for recognition and identification of objects, robot's TVS receives in real time a video stream containing an image of robot's working area (Figure 3).

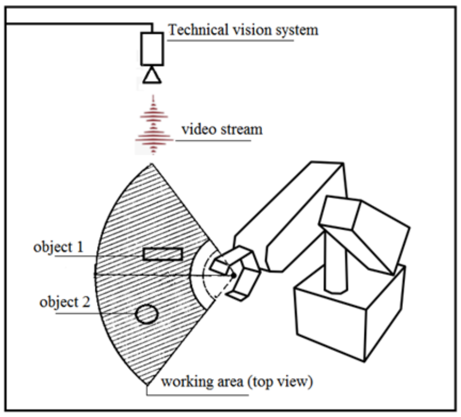


Figure 3. Schematic representation of specialized computer system

Then, within limits of working area of robot, it is necessary to identify objects of given classes, then – to determine their technical characteristics. In this case, recognition of robotization objects can be performed on basis of Haar classifiers trained in special way. Such classifiers have simple structure and it is possible, by successively applying them to an image, to find out how model works [25]. In this case, training of Haar classifiers should be carried out for each specific object.

At the same time, proposed development of classifier is based on machine learning algorithm of adaptive gain AdaBoost. The use of such an algorithm is due to fact that [26], [27]:

- flexibility, that is, ability to combine it with any machine learning algorithm, practically without setting parameters;
- versatility, since it can be used with numeric or text data;
- optimality in working with weak training algorithms, so such models can achieve an

accuracy much higher than random in solving classification problem;
- extensibility, that is, it can be extended to learning tasks more complex than binary classification.

The development of classifiers will consist of following stages:

- formation of learning sample;
- creation of positive samples;
- training of classifier;
- determination of its effectiveness.

Below, developed components of such stages are presented in more detail, which are submitted in form of specific command line utilities in Microsoft Visual Studio system in OpenCV language.

The initial stage is formation of learning sample.

The training set contains data on values of recognized objects features and classes corresponding to these objects. If amount of data is limited, then an analysis of its sufficiency is required to solve problem.

The samples contain groups of classes "bolt", "nut", "resistor", etc.

Figure 4 shows some of object recognition (OR) in real environment of robot workspace. The more similar sample will be to what will be recognized, better results will be.

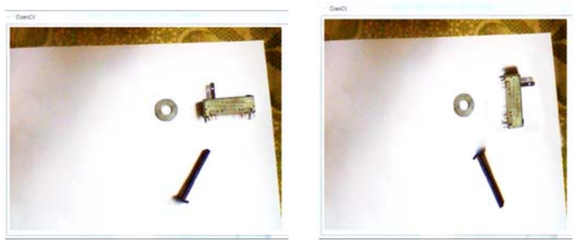


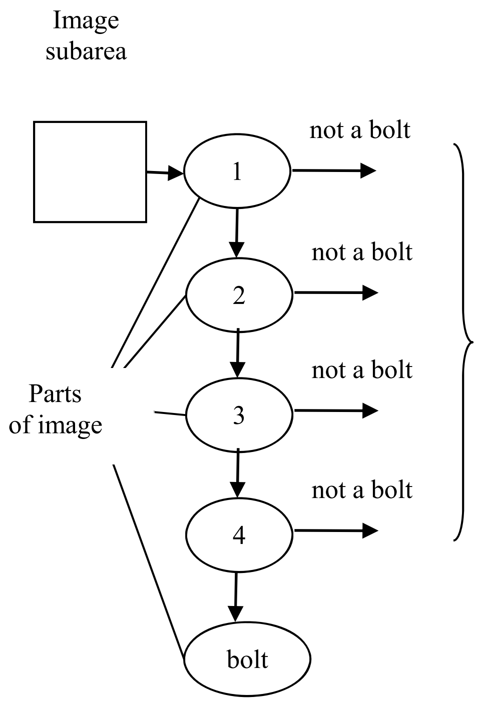
Figure 4. Some ORs in real robot workspace environment

Then, algorithm for forming a learning sample will contain following stages:

1) Using video camera, take series of shots containing recognition object in different positions on white background. Save received images in folder, for example, image.
2) Take series of different background options shots. Also save in folder like bg.
3) Create file describing location of target object in each frame, to do this, specify name of folder containing images, name of image with extension, separated by space, number of objects present in frame and coordinates of window containing object (image / 1.jpg 1 256 93 340 180). Save file with dat extension.
4) Create file and list in it images containing background (bg / frame1.jpg) and save with dat extension.
5) Save all resulting files, as well as “createsamples” and “haartraining” utilities in separate folder, for example, cascade.

In this work, “createsamples” utility was used to create an entire database of images from a single sample.

In Figure 5. architecture of one of classifiers is shown, since OR can be formalized by different types of characteristics.



In Figure 5, example of classifier architecture is considered, where 1, 2, 3, 4 are parts of images on which features are analyzed.

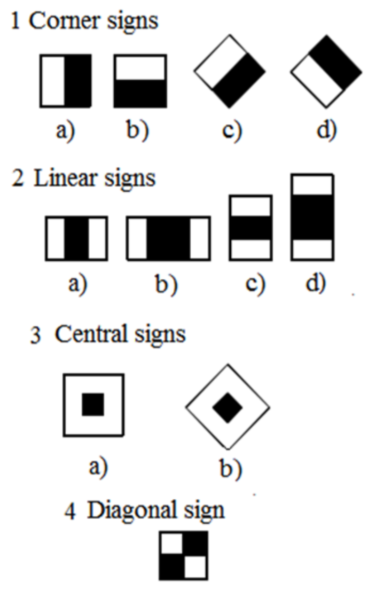
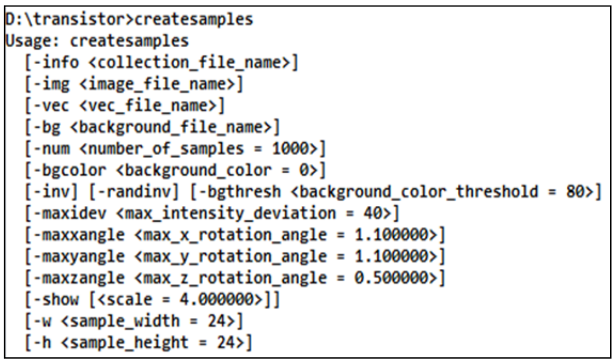


Figure 5. Example of classifier architecture

The next stage is creation of positive samples. Positive samples were also created using "createsamples" utility.

Set of positive samples can be created either from single image of object or from series of images in which windows containing target object are marked. From single image, by random transformations (rotation, color change, placement on different background, etc.), large set of samples for training is obtained.

An example of utility parameters set is shown in Figure 6, parameter values are set on command line (D:\ > transistor> createsamples).



```
D:\transistor>createsamples
Usage: createsamples
  [-info <collection_file_name>]
  [-img <image_file_name>]
  [-vec <vec_file_name>]
  [-bg <background_file_name>]
  [-num <number_of_samples = 1000>]
  [-bgcolor <background_color = 0>]
  [-inv] [-randinv] [-bgthresh <background_color_threshold = 80>]
  [-maxidev <max_intensity_deviation = 40>]
  [-maxxangle <max_x_rotation_angle = 1.100000>]
  [-maxyangle <max_y_rotation_angle = 1.100000>]
  [-maxzangle <max_z_rotation_angle = 0.500000>]
  [-show [<scale = 4.000000>]]
  [-w <sample_width = 24>]
  [-h <sample_height = 24>]
```

Figure 6. Parameters of "createsamples" utility

Basically there are four functions, option values of which can be different. The following parameters must be changed [28], [29]:

- <collection_file_name> – name of file with description of images series (for above case, this is info.dat file);
- img <image_file_name> – name of image containing target (for example, trans6.jpg);
- vec <vec_file_name> – name of output file containing positive training samples (for example, samples.vec);
- bg <background_file_name> – name of background description file (negatives.dat).

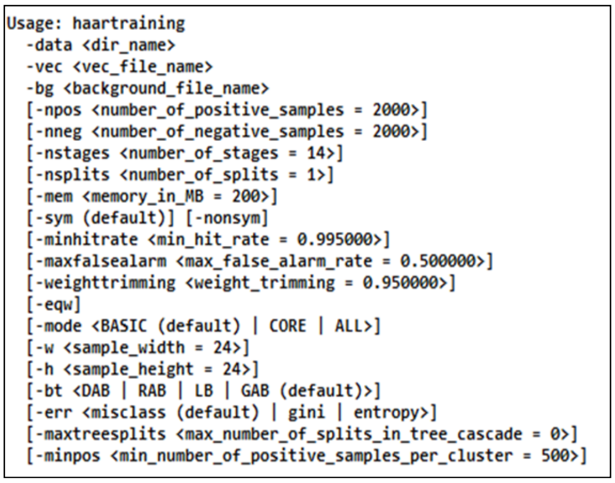
As a result, command line will look like this:

D:\ > transistor> createsamples -info info.dat -vec samples.vec -bg negatives.dat -num 1210 -w 24 -h 24.

Then stage follows – training of classifier.

The classifier was trained using haartraining () utility.

Example of utility parameters is shown in Figure 7.



```
Usage: haartraining
  -data <dir_name>
  -vec <vec_file_name>
  -bg <background_file_name>
  [-npos <number_of_positive_samples = 2000>]
  [-nneg <number_of_negative_samples = 2000>]
  [-nstages <number_of_stages = 14>]
  [-nsplits <number_of_splits = 1>]
  [-mem <memory_in_MB = 200>]
  [-sym (default)] [-nonsym]
  [-minhitrate <min_hit_rate = 0.995000>]
  [-maxfalsealarm <max_false_alarm_rate = 0.500000>]
  [-weighttrimming <weight_trimming = 0.950000>]
  [-eqw]
  [-mode <BASIC (default) | CORE | ALL>]
  [-w <sample_width = 24>]
  [-h <sample_height = 24>]
  [-bt <DAB | RAB | LB | GAB (default)>]
  [-err <misclass (default) | gini | entropy>]
  [-maxtreesplits <max_number_of_splits_in_tree_cascade = 0>]
  [-minpos <min_number_of_positive_samples_per_cluster = 500>]
```

Figure 7. Parameters of "haartraining" utility

The following is set of possible command line arguments:

- data <dir_name> – name of directory where classifier will be saved (data1);
- vec <vec_file_name> – name of output file containing positive training samples (vec1.cls);
- bg <background_file_name> – name of file with background description (bg.dat);
- npos <number_of_positive_samples>, nneg <number_of_negative_samples> – number of positive / negative samples used in training each level of classifier. Reasonable values: npos = 7000 and nneg = 3000;
- nstages <number_of_stages> – number of classifier levels for training (nstage = 14);
- nsplits <number_of_splits> – defines classifier to be used. If 1, then simple truncated classifier is used, if 2 or more then CART classifier with number_of_splits nodes is used;
- mem <memory_in_MB> – amount of available memory in MB. More memory – faster learning process;
- sym (default), nonsym – determines if target object has vertical axial symmetry (by default – yes), presence of symmetry speeds up learning process;
- minhitrate <min_hit_rate> – minimum hit rate for each classifier level. The overall hit rate can be roughly defined as min_hit_rate_number_of_stages;
- maxfalsealarm <max_false_alarm_rate> – maximum level of false positives for each classifier level. The overall false alarm rate can be roughly defined as max_false_alarm_rate_number_of_stages;
- weighttrimming <weight_trimming> – parameter determines whether and to what extent adjustment of weights will be used. The recommended value is about 0,90;

- w <sample_width>, h <sample_height> – width and height of samples used in training classifier.

Thus, command line can be represented as follows:

D:\ > transistor> haartraining -data haarcascade -vec samples.vec -bg negatives.dat -nstages 20 -nsplits 2 - -npos 7000 -nneg 3019 -w 20 -h 20 -nonsym -mem 512 -mode ALL

To determine efficiency of created classifier, performance utility is used (Figure 8).

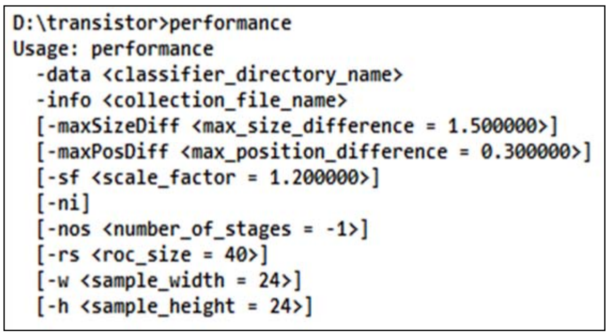
As arguments, function takes set of labeled images (images for which location of target object is known).

Then utility applies analyzed classifier to this set of images and gives out indicators of classifier efficiency. These indicators include number of objects found, number of misses, number of false positives, and other information [29].

The function takes following command line arguments:

- data <dir_name> – directory where trained classifier is stored;
- info <collection_file_name> – file with description of test samples;
- maxSizeDiff <max_size_difference> – defines requirements for accuracy of matching dimensions, and accuracy of matching centers of rectangles. The default is 1,2 and 0,3, respectively;
- sf <scale_factor> – detection parameter. Default is 1,2;
- w <sample_width>, h <sample_height> – sizes of samples used in training classifier.

The command line arguments must exactly match values used when running haartraining utility.



```
D:\transistor>performance
Usage: performance
  -data <classifier_directory_name>
  -info <collection_file_name>
  [-maxSizeDiff <max_size_difference = 1.500000>]
  [-maxPosDiff <max_position_difference = 0.300000>]
  [-sf <scale_factor = 1.200000>]
  [-ni]
  [-nos <number_of_stages = -1>]
  [-rs <roc_size = 40>]
  [-w <sample_width = 24>]
  [-h <sample_height = 24>]
```

Figure 8. Parameters of performance utility

Thus, command line might look like this:

D:\ > transistor> performance – data data1 –info info.dat – w 20 – h 20.

After testing cascade, resulting classifier (data1.xml) must be added to project and its name must be specified in load function cvHaarDetectObjects ().

The described methodology for development of classifiers has fairly wide application. It can be successfully combined with other algorithms. For example, described method can be used to find object in image, and semantic network or another method can be used for recognition.

4. Some Implementation Features of Object Recognition Procedure for Specialized Computer Systems

To start recognizing and identifying objects in robot's workspace using OpenCV computer vision library, we present a diagram of relationship between research elements, which is shown in Figure 9.

Information about objects in robot's workspace, thanks to use of WEB-camera connected to PC, will be processed using functions of OpenCV program. Identification of objects will occur in real time, thanks to the use of previously developed series of Haar classifiers for typical objects. Space modeling functions provide comparison of recognized objects with templates stored in memory and, based on them, establish semantic connections between objects, that is, build semantic network of workspace.

To implement image scanning process, you must specify rectangle that bounds image CvRect rect and number of neighboring rectangles in int neighbors group. Next, should connect Haar classifiers and indicate their respective names.

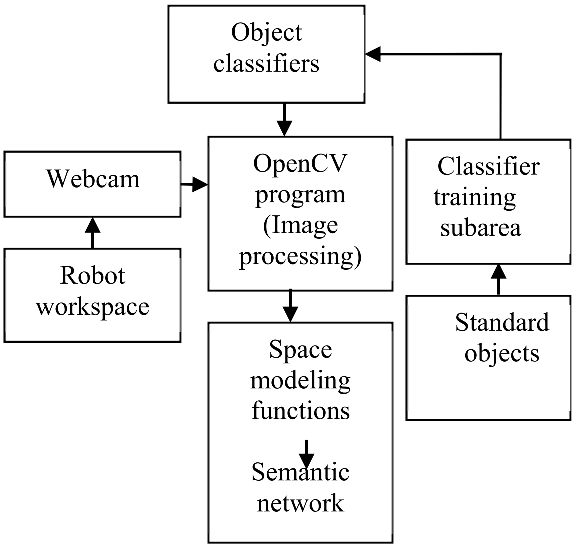


Figure 9. Diagram of research elements relationship

Then code fragment describing connection of classifiers will take following form:

```
bool bCreate=true;
static CvHaarClassifierCascade* cascade = 0;
static CvHaarClassifierCascade* cascade2 = 0;
static CvHaarClassifierCascade* cascade3 = 0;
const char* cascade_name = «data.xml»;
const char* cascade_name = «trans.xml»;
const char* cascade_name2 = «condensator.xml».
```

Also, in constructor of corresponding utility, we have to specify handle of main program window: HWND w = this–> GetSafeHwnd (), and also determine number of cameras using command: int ncams = cvcamGetCamerasCount ().

To organize output of visual information processing results, use cvNamedWindow ("result", 2) function, where 2 is window identifier. The result of each transformation of original image is displayed using cvShowImage () command. Image processing in OpenCV is done frame by frame, and callback function (mycallback), which in developed software looks like this: void mycallback (IplImage * src).

The cvHaarDetectObjects function finds rectangular areas in given image that most likely contain object against which classifiers are trained and returns them as part of sequence. The cvHaarDetectObjects function scans image multiple times at different zoom values. Parameters of cvHaarDetectObjects () function:

- image – variable identifying object;
- cascade – Haar classifier;
- storage – memory area for storing sequence of images;
- scale_factor – scaling factor (for example, 1.1 means a 10 % increase);
- min_neighbors – minimum number of neighboring rectangles (minus 1);
- flags – type of transaction$
- min_size – minimum window size, usually equal to size of training samples (~ 20x20).

Finding and identifying objects occurs thanks to cycle presented below:

A snippet of code describing cycle of processing image frames and identifying objects:

```
for( i = 0; i < (faces ? faces–>total : 0); i++ )
{CvRect* r = (CvRect*)cvGetSeqElem( faces, i );
CvPoint center;
int radius;
center.x = cvRound((r–>x + r–>width*0.5)*scale);
center.y = cvRound((r–>y + r–>height*0.5)*scale);
radius = cvRound((r–>width + r–>height)*0.25*scale);
```

Drawing functions are used to highlight recognized objects:

- circles: cvCircle(src, center, radius/2, CV_RGB(255,0,0), 3, 8, 0);
- rectangle: cvRectangle (src, cvPoint (center.x – radius/2, center.y – radius/2), cvPoint (center.x+radius/2,center.y+radius/2),CV_RGB(0,255,0)).

A whole array of points can also be used to construct desired shape.

To display name and coordinates of recognized object, use wsprintf (str, "Resistor (% d% d)", center.x, center.y) function, where str is output string, (% d% d) is output format, center. x, center.y – coordinates and size of text. An example of performing object selection is shown in Figure 10.

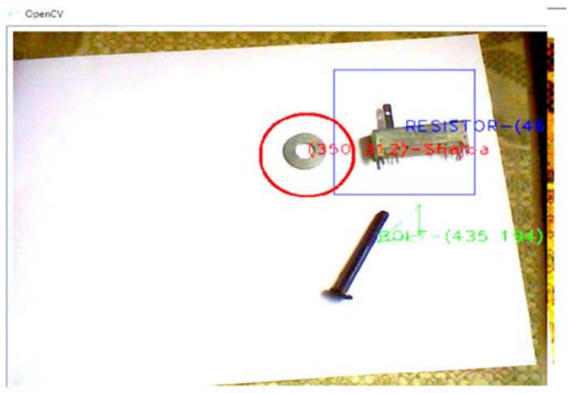


Figure 10. Real example of objects selection on test image

In the process of identifying objects in working space of robot, emergency situations may occur, leading to malfunction of program and disrupting process of object recognition. Such situations arise, for example, when used camera is inoperative, corresponding classifier is not connected, or when it does not find objects.

Control over exclusion of such situations is carried out as follows: in constructor of main window:

- function if (ncams) bCreate = true; – checks for connected cameras;
- function of connecting cascade if (cascade) – checks whether specified classifier is present in project,
- function for (i = 0; i <(faces? faces–> total: 0); i ++) – checks if classifier finds any objects at all.

5. Conclusion

In work, classifiers for recognizing robotization objects were created. Haar classifiers were used to process video information. The use of Haar classifiers gives good image quality when scaling and requires minimum amount of resources, which is important when solving problem of identifying robotization objects.

The development methodology is based on AdaBoost adaptive gain machine learning algorithm, as it will improve performance, and it also turns weak training algorithms into strong ones for solving classification problems.

Thus, classifiers created in this work for recognizing objects in specialized computer systems will make it possible to determine semantic connections between identified objects of workspace.

If semantic connections fit into certain scheme, then we can talk about possibility of robot performing any actions with objects according to this model, for example, mechanical assembly or assembly of radioelements.

Identification of objects in working space of robot will be carried out in real time, thanks to the use developed series of Haar classifiers for typical objects (bolt, nut, etc.).

To determine semantic links between identified objects of workspace, function of classifying objects according to established classes was developed in work.

References

[1]. Baker J. H., & et al.. (2021). Some Interesting Features of Semantic Model in Robotic Science. International Journal of Engineering Trends and Technology, 69(7), 38-44. https://doi.org/10.14445/22315381/IJETT-V69I7P205.

[2]. Alshorman, A. M., Alshorman, O., Irfan, M., Glowacz, A., Muhammad, F., & Caesarendra, W. (2020). Fuzzy-based fault-tolerant control for omnidirectional mobile robot. *Machines*, *8*(3), 55. https://doi.org/10.3390/machines8030055.

[3]. Useche, P., Jimenez-Moreno, R., & Baquero, J. M. (2020). Algorithm of detection, classification and gripping of occluded objects by CNN techniques and Haar classifiers. *International Journal of Electrical and Computer Engineering (IJECE)*, *10*(5), 4712-4720. https://doi.org/10.11591/ijece.v10i5.pp4712-4720

[4]. Srithar, S., Mary, I. M., Baskaran, P., & Maheswaran, T. (2021, May). Improved Haar Cascade Feature Extraction and Access Control Framework for Rich Internet Applications. In *Journal of Physics: Conference Series* (Vol. 1916, No. 1, p. 012019). IOP Publishing. https://doi.org/10.1088/1742-6596/1916/1/012019.

[5]. Hussain, N., Khan, M. A., Sharif, M., Khan, S. A., Albesher, A. A., Saba, T., & Armaghan, A. (2020). A deep neural network and classical features based scheme for objects recognition: an application for machine inspection. *Multimedia Tools and Applications*, 1-23. https://doi.org/10.1007/s11042-020-08852-3.

[6]. Almanza, C., Baquero, J. M., & Jiménez-Moreno, R. (2021). Robotic hex-nut sorting system with deep learning. *International Journal of Electrical and Computer Engineering (IJECE)*, *11*(4), 3575-3583. https://doi.org/10.11591/ijece.v11i4.pp3575-3583.

[7]. Jiménez-Moreno, Fonseca, A. R., Ramírez, J. L. (2020). Object gripping algorithm for robotic assistance by means of deep leaning. *International Journal of Electrical and Computer Engineering (IJECE)*, 10(6), 6292-6299. https://doi.org/10.11591/ijece.v10i6.pp6292-6299.

[8]. Pinzón-Arenas, J. O., & Jiménez-Moreno, R. (2020). Object sorting in an extended work area using collaborative robotics and DAG-CNN. *ARPN Journal of Engineering and Applied Sciences*, *15*(2).

[9]. Girshick, R., Donahue, J., Darrell, T., & Malik, J. (2014). Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 580-587). https://doi.org/10.1109/CVPR.2014.81.

[10]. Girshick, R. (2015). Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision* (pp. 1440-1448). https://doi.org/10.1109/ICCV.2015.169.

[11]. Ren, S., He, K., Girshick, R., & Sun, J. (2015). Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, *28*, 91-99. https://doi.org/10.1109/TPAMI.2016.2577031.

[12]. Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 779-788). https://doi.org/10.1109/CVPR.2016.91.

[13]. Jiménez-Moreno, R., Pinzón-Arenas, J. O., & Pachón-Suescún, C. G. (2020). Assistant robot through deep learning. *International Journal of Electrical and Computer Engineering*, *10*(1), 1053. https://doi.org/10.11591/ijece.v10i1.pp1053-1062.

[14]. Abdulrazzaq, M. B., & Saeed, J. N. (2019, April). A comparison of three classification algorithms for handwritten digit recognition. In *2019 International Conference on Advanced Science and Engineering (ICOASE)* (pp. 58-63). IEEE. https://doi.org/10.1109/ICOASE.2019.8723702.

[15]. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., ... & Fei-Fei, L. (2015). Imagenet large scale visual recognition challenge. *International journal of computer vision*, *115*(3), 211-252. https://doi.org/10.1007/s11263-015-0816-y.

[16]. He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 770-778). https://doi.org/10.1109/CVPR.2016.90.

[17]. LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *nature*, *521*(7553), 436-444. https://doi.org/10.1038/nature14539.

[18]. Liu, H., & Zhang, Y. (2019). Image-driven structural steel damage condition assessment method using deep learning algorithm. *Measurement*, *133*, 168-181. https://doi.org/10.1016/j.measurement.2018.09.081.

[19]. Fazilov, S., Mirzaev, O., Saliev, E., Khaydarova, M., Ibragimova, S., & Mirzaev, N. (2019, June). Model of recognition algorithms for objects specified as images. In *2019 9th International Conference on Advanced Computer Information Technologies (ACIT)* (pp. 479-482). IEEE. https://doi.org/10.1109/ACITT.2019.8779943.

[20]. Kherif, F., & Latypova, A. (2020). Principal component analysis. In *Machine Learning* (pp. 209-225). Academic Press. https://doi.org/10.1016/B978-0-12-815739-8.00012-2

[21]. Karamizadeh, S., Abdullah, S. M., Zamani, M., & Kherikhah, A. (2015). Pattern recognition techniques: studies on appropriate classifications. In *Advanced Computer and Communication Engineering Technology* (pp. 791-799). Springer, Cham. https://doi.org/10.1007/978-3-319-07674-4_74.

[22]. Li, P., Wang, D., Wang, L., & Lu, H. (2018). Deep visual tracking: Review and experimental comparison. *Pattern Recognition*, *76*, 323-338. https://doi.org/10.1016/j.patcog.2017.11.007

[23]. Torshin, I. Y., & Rudakov, K. V. (2016). On metric spaces arising during formalization of problems of recognition and classification. Part 2: density properties. *Pattern Recognition and Image Analysis*, *26*(3), 483-496. https://doi.org/10.1134/S1054661816030202.

[24]. Li, Y., Wang, G., Nie, L., Wang, Q., & Tan, W. (2018). Distance metric optimization driven convolutional neural network for age invariant face recognition. *Pattern Recognition*, *75*, 51-62. https://doi.org/10.1016/j.patcog.2017.10.015

[25]. Li, J., Zhao, Z., Liu, Y., Li, J., Cheng, Z., & Wang, X. (2017). A comparative study on machine classification model in lung cancer cases analysis. *International Journal of Applied Systemic Studies*, *7*(1-3), 13-29. https://doi.org/10.1504/IJASS.2017.088906

[26]. Ferreira, J. M., Pires, I. M., Marques, G., Garcia, N. M., Zdravevski, E., Lameski, P., ... & Spinsante, S. (2020). Identification of daily activites and environments based on the adaboost method using mobile device data: A systematic review. *Electronics*, *9*(1), 192. https://doi.org/10.3390/electronics9010192.

[27]. Howse, J. (2014, September). Training detectors and recognizers in Python and OpenCV. In *2014 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)* (pp. 1-2). IEEE Computer Society. https://doi.org/10.1109/ISMAR.2014.6948516.

[28]. Soo, S. (2014). Object detection using Haar-cascade Classifier. *Institute of Computer Science, University of Tartu*, *2*(3), 1-12.

[29]. Shanahan, J. G., & Dai, L. (2020, August). Introduction to Computer Vision and Real Time Deep Learning-based Object Detection. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining* (pp. 3523-3524). https://doi.org/10.1145/3394486.3406713