International Journal of Engineering

Journal Homepage: www.ije.ir

Fast Color Straight Line Pattern Recognition in an Object using High Speed Self Learning Devices

Rasiq S. M.*, S. Krishnakumar

School of Technology and Applied Sciences, Mahatma Gandhi University Regional Centre, Edappally, Kochi, India

PAPER INFO

Paper history:
Received 23 November 2020
Received in revised form 08 Februery 2021
Accepted 03 March 2021

Keywords:
Rasiq Krishnakumar Algorithm
Rasiq Krishnakumar Device
Fast Object Recognition
Color Straight Line Pattern

ABSTRACT

Most of the man-made objects are having some straight lines with colors. A very high-speed object recognition method using color straight line patterns is carried out using a novel self-learning device: Rasiq Krishnakumar Device(RKD). Instead of that Artificial Neural Networks (ANN), RKD based networks are used for different steps in this pattern classification. The color and straight-line features are extracted by using high-speed color segmentation and fast efficient straight-line segment extraction methods using the RKD based systems. The training and the testing algorithms of the pattern classification are using RKD-based processing. The fast color features extraction method uses an array of RKD-based devices and the fast efficient straight-line segment extraction method employs an array of processing elements and a main control unit. Some fusion devices are used for a straight line with colors features. The area of interest and the area of line segments of a particular object are other features for improving the accuracy of object classification.

doi: *10.5829/ije.2021.34.05b.15*

NOMENCLATURE

c	Common colors	*C*	Common color ratios
lc	Colors related to staight lines	*LC*	Color count ratios
O	Set of objects	*k and k'*	Known values
a_1	Area of region of interest	a_2	Area of line segments

1. INTRODUCTION

An RKD is a self-learning device. RKD networks are used for multi-class pattern recognition problems, fast efficient straight line detection, and color based image segmentation [1-3]. The applications of high-speed object recognition methods are mainly included in robotics, driverless vehicles, and artificial intelligence. The difficulties in object recognition under various circumstances are lightning conditions, the position of the object in the image, image rotation, occlusion, and change in the size of the object [4]. The other main difficulties in a high-speed object recognition system are the hardware complexity and the processing time [5]. The methods carried out are optimized for these difficulties.

The main reason for the hardware complexity and the high processing time is the complications in ANN for training and testing. Convolutional Neural Networks (CNN) are used for object recognition applications that need sophisticated hardware for high-speed applications. The performance of CNN increases when the number of convolutional layers, pooling layers, and fully connected layers increases [6]. The increase in layers causes complications in training and requires large training data [7]. Sometimes these approaches are not suitable for high-speed object recognition applications. The sophisticated processes in CNN methods for training, reduce the scope for further processing of erroneous predictions. Unobservable processes in the hidden layers of ANN are another challenge for further processing and the weight values in the hidden layers cannot be edited [8].

*Corresponding Author Institutional Email: rasiqsm2@gmail.com (Rasiq S. M.)

If a single weight in the trained ANN is edited, the stability of the entire network is affected [9]. In the case of RKD networks, the knowledge acquired by the network can be edited for further applications.

This work uses a novel method for training and testing based on RKD. The RKD is simple, fast, required fewer hardware components and training samples, and having easily accessible processing variables [10]. Three RKDs are formed to a basic color matching device (BCMD) used for learning as well as for detecting a particular color. There is n number of BCMDs used for learning n number of colors of a multiple colored object. An array of BCMDs is needed for segmenting these n number of colors. The BCMD array segments a single color at a time. All colors in the object are recognized after n number of segmentation [3].

The straight-line segment extraction method requires a smaller number of processing elements (PE) compared to a parallel Hough transform (HT) method. The speed and accuracy of straight-line extraction are very high compared to the parallel HT method [2]. Straight lines with colors (SLC) are formed by combining the color recognition method and parallel straight-line segment extraction method. The main methods used here are recognition of multiple colors, straight-line extraction, straight lines with colors recognition, the object area, and the line segments area.

For fast processing, six two-dimensional arrays and a straight line color fusion device are proposed. A few straight line color fusion devices process the six arrays results in fast processing. A significant property of the RKD network is that comparatively higher classification accuracy of 92.3% is obtained with a few training images. Moreover, the processing time and memory requirements are less compared to the ANN methods.

Section 2 discusses the theory and methodology which includes recognition of multiple colors, high-speed straight-line extraction, six arrays for SLC data extraction, straight-line color fusion device, SLC calculations, and multi-class pattern recognition using RKD array. Session 3 deals with results and discussion, and finally session 4 includes the conclusion.

2. THEORY AND METHODOLOGY

A multi-class pattern recognition method is carried out by using an RKD array, which requires color features, SLC features, and the object area. BCMDs are used for learning and detecting color features. Fast efficient straight line extraction method is combined with the color recognition method using six arrays of SLC data extraction. SLC features are obtained by straight line color fusion devices. The object is recognized by multi-class pattern recognition using an RKD array.

2. 1. Recognition of Multiple Colors The BCMD is capable of learning one color in an image at a time. The system can identify multiple colors present in the object by training a number of BCMDs [3]. A single color is learned by a BCMD by selecting some sample points in the colored region of the image. The number of BCMDs needed for learning is n, for n number of colors in the object. The learned colors are represented as color 1 (c_1), color 2 (c_2), color 3 (c_3) ,… , color n(c_n) by the system. Segmentation of a single color from an N x N image needs N^2 number of BCMDs. These BCMDs are arranged like an N x N array and each BCMD has stored the color attributes from the database. BCMDs check every pixel in the image and detect the colored region in the image. The system can recognize n number of colored regions of an object by checking in the image n times using the BCMD array.

2. 2. High-speed Straight-line Extraction The straight-line segment extraction method used here is very fast and efficient and it can extract straight lines within a few nanoseconds [2]. It is calculated that for an N x N image $6N^2$-16N +14 line alignments are possible. Each of these line alignments is connected to a PE. By using this method, hardware complexity is comparatively very much less than the parallel HT method. The least number of PEs required for the parallel HT method is MN^2, the value of M is the angular variations and N depends on the resolution of the image [11]. Those PEs perform arithmetic operations, cosine and sine value calculations, and reconfiguration of the PE mesh. In this method each PE performs pixel scanning in the line alignment, comparing line lengths with nearby PEs and eliminating errors. Parallel HT methods use less resolution line extraction. The method discussed in literature [2] is fast and accurate than parallel HT.

2. 3. Six Arrays for SLC Data Extraction The most processing time required step is SLC data extraction. In order to reduce the time of processing a parallel processing method is employed. For extracting SLC information six arrays are needed as shown in Figure 1. These arrays are having a size N x N. First array stores a binary image in which each element is having one-bit memory and it is split into several line alignments as described in literature [2]. Each element in the second and third array are having word length Log_2 (N), these arrays are x and y coordinates respectively, and are accessible by a PE used for fast line detection. The remaining three arrays are used for storing primary color information and these arrays are split into different line alignments like the first array, for the line extraction. These line alignments can be enabled. By simultaneous and synchronous scanning through these line alignments, the system can evaluate the color data of a line. An actual line has six- imaginary lines as shown in Figure 2. Two-

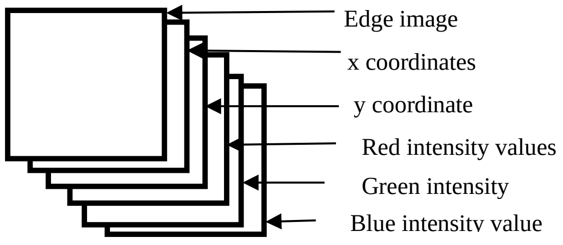


Figure 1. Six arrays for lc data extraction

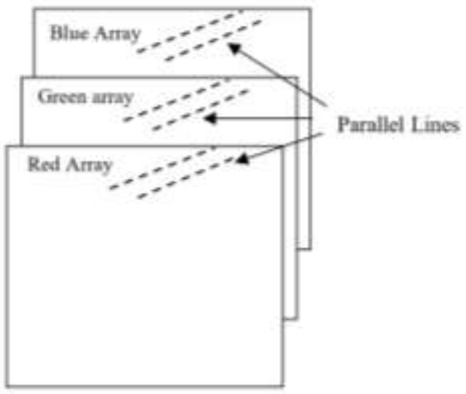


Figure 2. Parallel imaginary straight lines

lines in the red intensity array, two lines in the green intensity array, and two lines in the blue intensity array. Each element in these arrays has a one-byte word length. If these three arrays are combined, the original image is obtained. Simultaneous and synchronous scanning is needed for acquiring actual SLC data of a line.

2. 4. Straight Line Color Fusion Device

Different functions of a line color fusion device are as follows. At first, it acquires the line data from MCU [2] which includes the starting and ending points of a line and its identity in the first layer. Using the line data, the line color fusion device identifies and enables six parallel imaginary lines in the line alignments of fourth, fifth and sixth arrays, the starting and ending point data are used for segmenting the line alignments. Then the fusion device scans through the parallel imaginary lines in the above three layers. This scanning is simultaneous and synchronous inside the three layers. For n number of colors, there are 2 × n number of digital counters inside the line color fusion device and a digital counter count the number of pixels represents a color. There is a 2 × n number of BCMDs for learning/matching colors in the imaginary line segments. For matching, color data are stored from the database. A BCMD has three inputs, the number of imaginary line segments is six for a particular line segment and the number of colors is n. So, the number of BCMD required for a line color fusion device is 2×n. Each input of a BCMD match one primary color of an array mentioned above and each line segment has two imaginary lines in the array.

The number of line color fusion devices are used for scanning the imaginary parallel line segments is m, they scan through the line alignments of the above arrays. The line alignments are enabled by the line color fusion devices corresponding to the data from the MCU.

The line color fusion device enables corresponding starting and ending points in the line alignments in the above arrays. Then a fusion device scans through corresponding arrays' imaginary line segments and it finds the color information lc. A maximum number of fusion devices are equal to the maximum number of line segments receive from the MCU.

The scanning through imaginary parallel line segments is one of the most processing time requiring steps by the system, that is performed by the fusion device. The MCU collects all SLC data for evaluating the color count ratios.

2. 5. SLC Calculations

To identify the straight lines with color patterns, initially, the straight-line segments present in the image are extracted. The edge image is extracted by using parallel differentiators. Then all the line segments are extracted as described in literature [2]. The MCU provides the data of line segments present in a color image. Since these line segments are extracted from edges they do not give accurate colors. Edges are sharp changing pixels intensity values. So, the system draws two parallel imaginary straight line segments with each line segment. An actual straight-line segment should be in the middle of these parallel imaginary straight lines. The parallel imaginary straight lines are h pixels away from the actual straight-line segment; let the value of h be 10. These parallel imaginary straight lines contain color information related to that particular straight-line segment. The system scans through these parallel imaginary straight lines for finding the colors related to those particular straight-line segments. Thus, m number of the straight lines are having n number of color information represented as lc.

$$lc = \{ l_1c_1, l_1c_2, \dots, l_1c_n;\ \dots; l_2c_1, l_2c_2, \dots, l_2c_n;\ ..; l_mc_1, l_mc_2, \dots, l_mc_n\} \quad (1)$$

where m is the number of line segments and n is the number of colors.

The system learns to classify objects using a training algorithm. After learning is completed the system is tested using a testing algorithm. In the training as well as the testing stage, the system has to learn multiple colors present in the objects. For a class of objects, initially, the system learns the common colors present in that class represented as c, where c = {c_1, c_2, c_3, ..., c_n}, (n is the maximum number of common colors present in the object class).

$$C = c\ /\ \sum_{i=1}^{n} c_i \quad (2)$$

Then the system extracts straight lines and evaluates color count ratios LC.

The ratio of the number of pixels for a particular color of a straight-line segment is divided by the number of all the color pixels detected by different straight-line segments is called a color count ratio LC.

$$LC = lc \,/ \sum_{i=1}^{m} \sum_{j=1}^{n} l_i c_j \qquad (3)$$

where i = 1,2,3,.,.m and j = 1,2,3,..,n

The main intention of finding the color count ratio and C is due to the different sizes of objects in the image of the same class. If the object size is large, c and lc values are also large similarly if the object size in the image is small, c and lc values are also small. When C and the color count ratios are calculated they are almost the same for different sizes of objects in the image for a particular class.

Now each class of the objects represented as O_k is having the following attributes.

$O_k = \{C,\ LC\}$, such that $O_k \in O$, where k=1,2, 3,…,p (p is the number of objects), and O is the set of entire objects.

$C = \{C_1, C_2, C_3, \dots, C_n\}$, where n is the number of colors

$LC = \{L_1C_1, L_1C_2, \dots, L_1C_n;\ \dots; L_2C_1, L_2C_2, \dots, L_2C_n; \dots; L_mC_1, L_mC_2, \dots, L_mC_n\}$

where m is the number of lines and n is the number of colors RK algorithm converts each attribute as a low value and a high value.

2. 6. Multi-class Pattern Recognition Using RKD Array

Features X_1, X_2, X_3, …., X_d be in d dimensional feature space and which are to be classified into K classes. An RKD array having d number of RKDs and the outputs of these RKDs are ANDed using an AND gate can be used for classifying a particular class as shown in Figure 3.

RKD1 learns attribute values of X_1 which are k_1 and k_1', RKD2 learns attribute values of X_2 which are k_2 and k_2' similarly all RKDs produce their attribute values from corresponding feature variable after training. Thus, for a class O_k a set of attribute values are learned by the RKD array.

For recognizing or classifying a particular object, its attribute values are stored in the RKD array, then the feature vector is given as input to the RKD array, and the RKD array matches the feature vector with attribute values, if it is matched the object is recognized.

The RKD array is used in multi-class pattern recognition problems for high-speed object recognition applications. At a time, a d dimensional array learns to classify a single class of objects. After learning is completed the attribute values produced by the RKD array are stored in the database. Then the same RKD array can be used for learning another class of the object and the attribute values are stored again in the database.

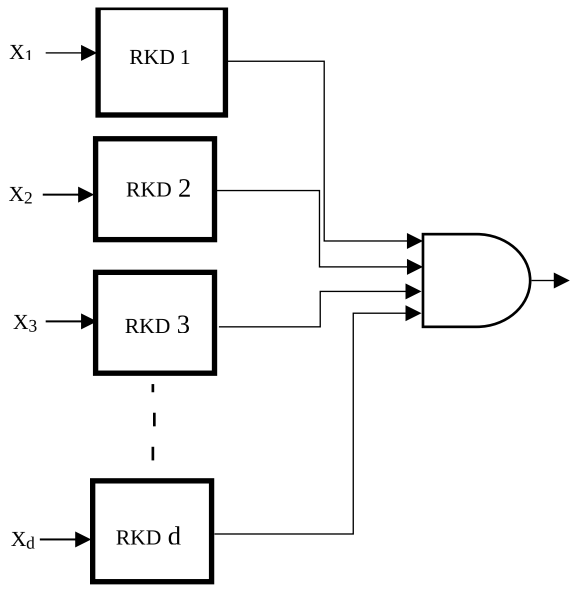


Figure 3. RKD array ANDed for classification

These processes are repeated p number of times for learning to classify p classes of objects.

While testing the input feature vector is given to the RKD array and the attribute values of each class are stored into the RKD array one by one. Let us suppose for a particular class all RKD outputs become high and the AND gate output becomes high then that class is said to be recognized.

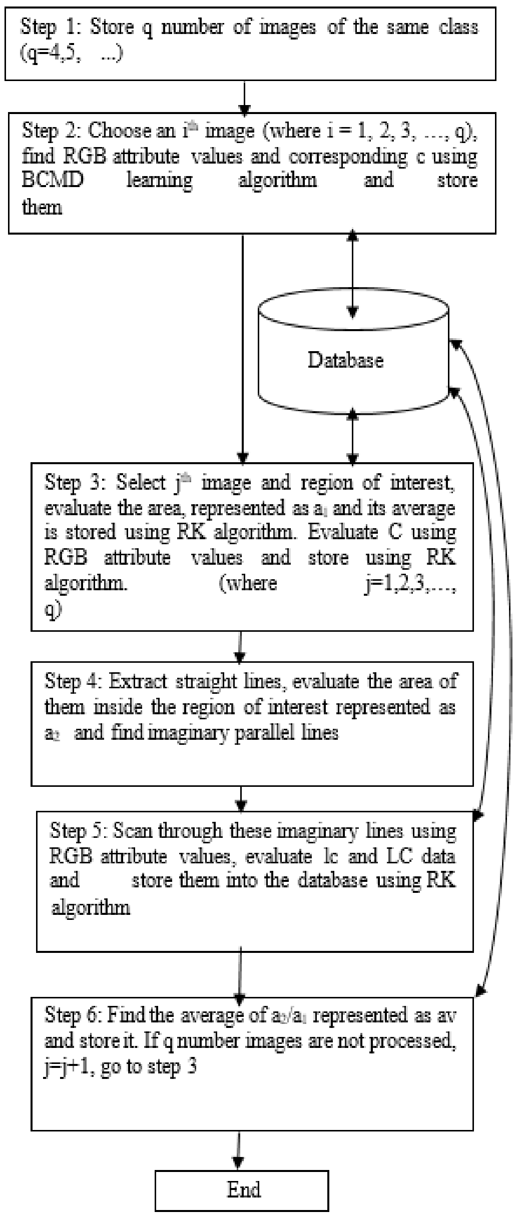
This method is very fast and it requires comparatively a few hardware components. It doesn't need the statistical approach of recognition or neural network approach. So, it is simple and requires less complicated processing. For straight-line extraction of an N x N image, each PE has to perform in three arrays of N x N size, because the size of the image is N x N. First array contains the edge image, the second array contains the coordinate x values, and the third array contains the y values. For pattern recognition, three additional arrays are needed.

3. RESULTS AND DISCUSSION

Table 1 shows the accuracies of various methods of object recognition from RGB images. The data sets used are the Washington RGB-D object dataset shown in Figure 4. The accuracy is calculated by averaging over classes and the presented method got an accuracy of 92.3%.

In this work speed of object recognition is in the nanosecond range. The system can process raw images directly, with no need for noise removal. The color segmentation is optimized by using BCMDs based on RKD, it is very fast and accurate. The color segmentation part needs lesser time compared to the straight-line segment extraction method. By using the HT method some lines may not be detected. The accuracy of the HT depends on accumulator cells and the bin size [2]. The proposed line segment extraction method extracts every line present in the image with higher accuracy and lesser hardware than parallel HT. The training and testing algorithms are also simple compared to other methods. Most of the works in this area require a large number of images for training. It is due to the fast learning property of RKDs, only a few images are needed for training the presented system.

Training Algorithm



Step 1: Store q number of images of the same class (q=4,5, ...)

Step 2: Choose an i^{th} image (where i = 1, 2, 3, ..., q), find RGB attribute values and corresponding c using BCMD learning algorithm and store them

Step 3: Select j^{th} image and region of interest, evaluate the area, represented as a_1 and its average is stored using RK algorithm. Evaluate C using RGB attribute values and store using RK algorithm. (where j=1,2,3,..., q)

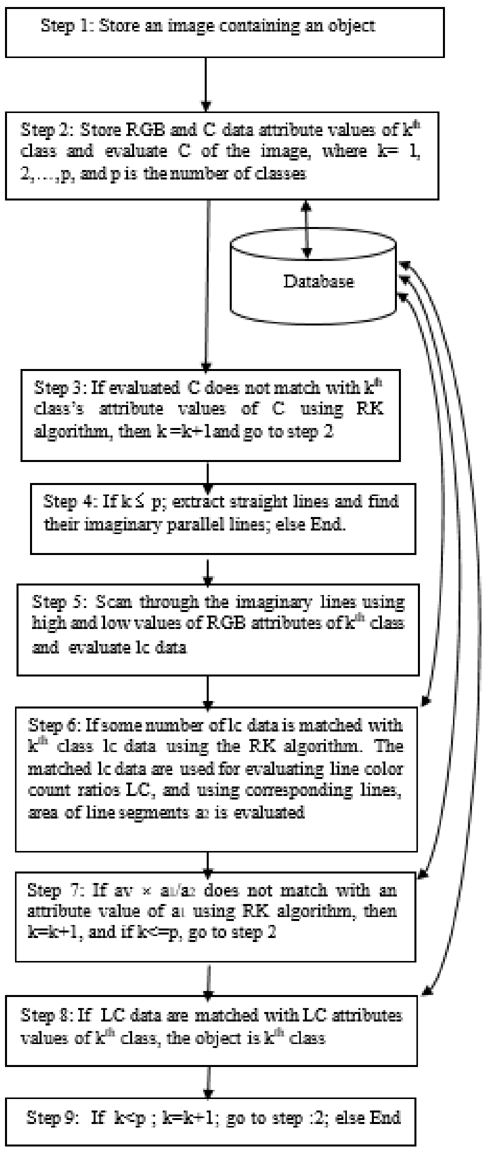
Step 4: Extract straight lines, evaluate the area of them inside the region of interest represented as a_2 and find imaginary parallel lines

Step 5: Scan through these imaginary lines using RGB attribute values, evaluate lc and LC data and store them into the database using RK algorithm

Step 6: Find the average of a_2/a_1 represented as av and store it. If q number images are not processed, j=j+1, go to step 3

End

Testing Algorithm



Step 1: Store an image containing an object

Step 2: Store RGB and C data attribute values of k^{th} class and evaluate C of the image, where k= 1, 2,...,p, and p is the number of classes

Step 3: If evaluated C does not match with k^{th} class's attribute values of C using RK algorithm, then k=k+1and go to step 2

Step 4: If $k \leq p$; extract straight lines and find their imaginary parallel lines; else End.

Step 5: Scan through the imaginary lines using high and low values of RGB attributes of k^{th} class and evaluate lc data

Step 6: If some number of lc data is matched with k^{th} class lc data using the RK algorithm. The matched lc data are used for evaluating line color count ratios LC, and using corresponding lines, area of line segments a_2 is evaluated

Step 7: If av × a_1/a_2 does not match with an attribute value of a_1 using RK algorithm, then k=k+1, and if k<=p, go to step 2

Step 8: If LC data are matched with LC attributes values of k^{th} class, the object is k^{th} class

Step 9: If k<p ; k=k+1; go to step :2; else End

The color noise can be reduced by using the smoothening technique in the learning step. While teaching a color, it is selected a number of points having the same color, then the device learns the color of that area. By adding a smoothening technique, a sharp variation in colors at some points must be avoided because they are considered as noise pixels. The smoothening does not affect the testing algorithm. In the teaching step, smoothening uses additional computation time. The smoothening technique is not required at the recognition algorithm. It is because the recognition algorithm recognizes only the color in a particular range learned by a BCMD.

TABLE 1. Comparison with state-of-the-art methods on the Washington RGB-D object dataset

Method	**Accuracy in %**
Linear SVM [13]	74.3 ± 3.3
kSVM [13]	74.5 ± 3.1
HKDES [14]	76.1 ± 2.2
Kernel Descripto [15]	77.7 ± 1.9
CNN-RNN [16]	80.8 ± 4.2
RGB-D HMP [17]	82.4 ± 3.1
MMSS [18]	74.6 ± 2.9
Fus-CNN (HHA) [19]	84.1 ± 2.7
Fus-CNN (Jet) [19]	84.1 ± 2.7
CFK [20]	86.8 ± 2.2
MDCNN [21]	87.9 ± 2.0
VGGnet + 3D CNN + VGG3D [22]	88.9 ± 2.1
RGB CNN+SVM[23]	87.5 ± 2.1
Presented Method	**92.3 ± 2.5**

CNN-RNN method requires the processing of a large number of image frames [12] for better accuracy. It is identified that the accuracy percentage increases almost proportional to the logarithm of the number of image frames processed. The batch size, as well as the number of epochs, are large. CNN methods require hundreds of epochs for better accuracy.

Figure 5 shows the accuracy percentage change in the RKD network method. Only a few numbers of image frames are processed. Accuracy percentage change increases almost proportional to the exponential of the number of training images and requires only a few numbers of training images for better results. Fast learning property is due to RK algorithm-based learning.

The lower values of LC data are known as k values, and higher values of LC data are known as k′ values. Most of the k and k′ ranges of each variable are not overlapping. If k─k′ values of a single variable in a class is different, the class is fully separated. Here, the classes are almost fully separated or mutually exclusive, and accurate classification results are obtained.

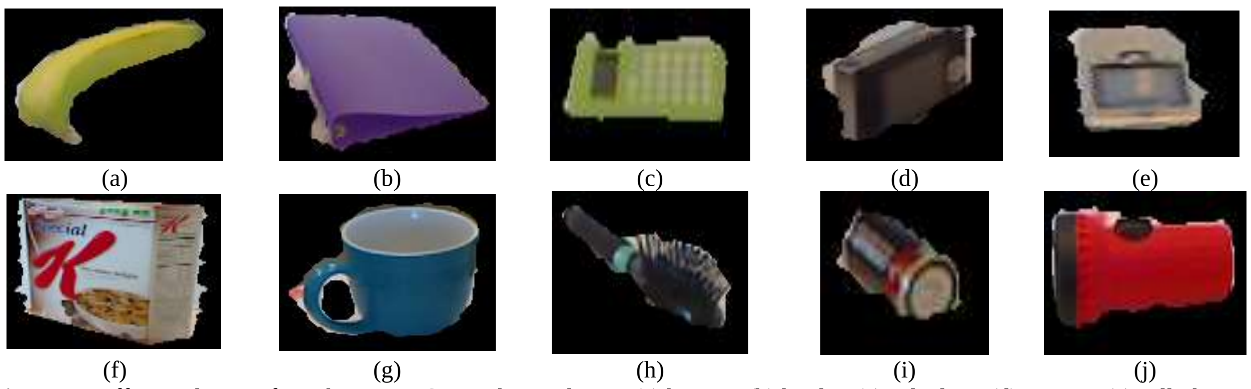


Figure 4. Different classes of Washington RGB D objects dataset (a) banana, (b) binder, (c) calculator,(d) camera, (e) cell phone, (f) cereal_box, (g) coffee mug, (h) comb, (i) dry battery and (j) flashlight

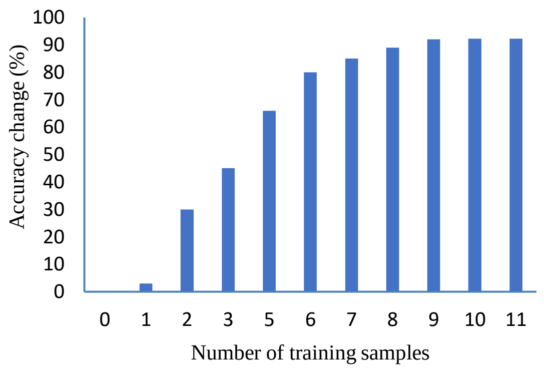


Figure 5. Change in classification accuracy with the number of training samples of the presented method

The accuracy of object recognition is estimated as 92.3% with a standard deviation of 2.5 when 10 classes of objects having different straight lines and color patterns are used. The number of training images used for each class is only 11 and tested with 200 images having 20 images with each class and obtained an accurate result for 185 testing images. The accuracy is averaged over classes. Figure 5 shows the accuracy change with respect to the number of training images. The accuracy change remains almost constant when the number of images is around 10. It is due to the limitations of the dataset. The color and SLC patterns of these classes have variations among them; in other words, the patterns of these classes are mutually exclusive. More accurate results compared to other works are obtained.

TABLE 2. Comparison of the number of computations and memory requirements for the different methods

Model	FLOPs	Bytes
AlexNet	7.29e+8	2.44e+8
CaffeNet	7.27e+8	2.44e+8
CNN-S	2.94e+9	4.12e+8
VGG-16	1.55e+10	5.53e+8
Presented	**7.07e+7**	**3.14e+6**

The computation time relies on the number of computations per frame. Table 2 shows the number of computations and memory requirements for the presented method. Which are significantly less than different neural network methods for a 224 x 224 image [24]. The number of computations and memory requirements are calculated by adding entire counts of different functioning devices and it is verified by simulations. A FLOP is the number of floating-point operations required to classify one image with the convolutional network. In the presented method, most of the calculations are using one-byte integers.

4. CONCLUSION

The present method is used for color and straight-line pattern recognition. The object recognition became very accurate by using features of line segments and colors. The color and SLC are recognized by separate parts, so it recognizes an object at a very high speed. The hardware complexity is very less compared to other methods. The RK algorithm reduces the hardware complexity, processing time, and memory requirements. The RK algorithm-based learning and recognition are faster than other methods. It is a logical device and the parameters are editable and it requires a few images for training. The method used for straight-line segment extraction is highly accurate compared to the parallel HT method. The line extraction algorithm uses comparatively little hardware complexity. The color pattern recognition unit also requires comparatively lesser hardware complexity. So, hardware implementation of this system will give better results.

5. REFERENCES

1. Rasiq S.M and S. Krishnakumar, "A Fast-Efficient Multi Class Pattern Recognition Method***", International Journal of Innovative Technology and Exploring Engineering*** ,Vol. 8, (2019), 1935-1938, doi: 10.35940/ijitee.L2893.1081219 9.
2. Rasiq S M, Jeevan K M and S Krishnakumar, "A Fast-Efficient parallel processing algorithm for straight line detection***", Indonesian Journal of Electrical Engineering and Computer Science***, Vol. 16, No. 3, (2019), 1320-1326, doi: 10.11591/ijeecs.v16.i3.pp1320-1326.
3. Rasiq S.M and S. Krishnakumar, "Parallel Processing Technique for Multiple Color Object Recognition", ***International Journal of Pure and Applied Mathematics***, Vol. 118, No.7, (2018), 125-130.
4. Sukanya C.M., Roopa Gokul, Vince paul, "A survey on object recognition method", ***International Journal of Computer Science & Engineering Technology***, Vol. 6, (2016), 48-52.
5. D. Maturana and S. Scherer, "VoxNet: A 3D Convolutional Neural Network for real-time object recognition", IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Hamburg, Germany, (2015), 922-928, doi: 10.1109/IROS.2015.7353481
6. N. Passalis and A. Tefas, "Training Lightweight Deep Convolutional Neural Networks Using Bag-of-Features Pooling", ***IEEE Transactions on Neural Networks and Learning Systems***, Vol. 30, No. 6, (2019), 1705-1715, doi: 10.1109/TNNLS.2018.2872995.
7. M. Kim, J. H. Jung, J. U. Ko, H. B. Kong, J. Lee and B. D. Youn, "Direct Connection-Based Convolutional Neural Network (DC-CNN) for Fault Diagnosis of Rotor Systems", ***IEEE Access***, Vol. 8, (2020), 172043-172056, doi: 10.1109/ACCESS.2020.3024544.
8. Li J, Chen BM, Lee GH, "So-net: Self-organizing network for point cloud analysis", Proceedings of the IEEE conference on computer vision and pattern recognition, (2018), 9397-9406, doi: 10.1109/CVPR.2018.00979.
9. X. Guo, J. Wang, F. Liao and R. S. H. Teo, "CNN-Based Distributed Adaptive Control for Vehicle-Following Platoon With Input Saturation", ***IEEE Transactions on Intelligent Transportation Systems,*** Vol. 19, No. 10, (2018), 3121-3132, doi: 10.1109/TITS.2017.2772306.
10. Rasiq S.M., S.Krishnakumar, "Parallel Processing Technique for High Speed Object Recognition", ***International Journal of Computer Applications***, Vol. 99, No. 4, (2014), 23-27, doi: 10.5120/17361-7874 .
11. Ling Chen and Hongjia Chen, "A fast efficient parallel Hough transform algorithm on LARPBS", ***The Journal of Supercomputing***, Vol. 29, (2004), 185-195, doi 10.1023/B:SUPE.0000026850.06646.3c.
12. C. Wang, M. Cheng, F. Sohel, M. Bennamoun, J. Li, "NormalNet: a voxel-based CNN for 3D object classification and retrieval", ***Neurocomputing,*** Vol. 323, (2019), 139-147 doi: 10.1016/j.neucom.2018.09.075.
13. Lai K., Bo L., Ren X., Fox D., " A large-scale hierarchical multi-view RGB-D object dataset"; Proceedings of the IEEE International Conference on Robotics and Automation; Shanghai, China, (2011), 1817–1824,doi: 10.1109/ICRA.2011.5980382.
14. Girshick R., Donahue J., Darrell T., Malik J., "Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation", Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition; Columbus, OH, USA. (2014), 580-587, doi: 10.1109/CVPR.2014.81.
15. Bo L., Ren X., Fox D., "Depth kernel descriptors for object recognition", Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems; San Francisco, CA, USA. (2011), 821-826, doi:10.1109/IROS.2011.6095119.
16. Socher R., Huval B., Bhat B., Manning C.D., Ng A.Y., "Convolutional-Recursive Deep Learning for 3D Object Classification***"***, Proceedings of the International Conference on Neural Information Processing Systems; Lake Tahoe, NV, USA, (2012), 656-664.
17. Bo L., Ren X., Fox D., "Unsupervised Feature Learning for RGB-D Based Object Recognition", Proceedings of the International

Symposium on Experimental Robotics; Québec City, QC, Canada, (2012), 387–402.

18. Wang A., Cai J., Lu J., Cham T.J., "MMSS: Multi-modal Sharable and Specific Feature Learning for RGB-D Object Recognition", Proceedings of the IEEE International Conference on Computer Vision; Santiago, Chile, (2015); 1125-1133, doi: 10.1109/ICCV.2015.134.
19. Eitel A., Springenberg J.T., Spinello L., Riedmiller M., Burgard W., "Multimodal Deep Learning for Robust RGB-D Object Recognition", Proceedings of the IEEE/RSJ International Conference on Intelligence Robots and Systems; Hamburg, Germany, (2015), 681-687, doi: 10.1109/IROS.2015.7353446.
20. Cheng Y., Cai R., Zhao X., Huang K., "Convolutional Fisher Kernels for RGB-D Object Recognition", Proceedings of the International Conference on 3D Vision; Lyon, France, (2015), 135-143, doi: 10.1109/3DV.2015.23.
21. Rahman M.M., Tan Y., Xue J., Lu K., " RGB-D object recognition with multimodal deep convolutional neural networks", Proceedings of the IEEE International Conference on Multimedia and Expo; Hong Kong, China, (2017), 991-996, doi: 10.1109/ICME.2017.8019538.
22. Zia S., Yüksel B., Yüret D., Yemez Y., "RGB-D Object Recognition Using Deep Convolutional Neural Networks", Proceedings of the IEEE International Conference on Computer Vision Workshops; Venice, Italy, (2017), 887-894, doi: 10.1109/ICCVW.2017.109.
23. Zeng H, Yang B, Wang X, Liu J, Fu D., "RGB-D Object Recognition Using Multi-Modal Deep Neural Network and DS Evidence Theory", ***Sensors (Basel),*** Vol. 19, (2019), 1-19, doi: 10.3390/s19030529 .
24. J. Wu, C. Leng, Y. Wang, Q. Hu and J. Cheng, "Quantized Convolutional Neural Networks for Mobile Devices", IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, (2016), 4820-4828, doi: 10.1109/CVPR.2016.521.

Persian Abstract

چکیده

بیشتر اشیا–ساخته شده توسط انسان دارای برخی از خطوط مستقیم با رنگ هستند. یک روش شناسایی سریع شی با استفاده از الگوهای خط مستقیم رنگ با استفاده از یک دستگاه خودآموز جدید انجام می شود: دستگاه .Rasiq Krishnakumar (RKD)به جای شبکه های عصبی مصنوعی (ANN)، شبکه های مبتنی بر RKD برای مراحل مختلف در این طبقه بندی الگو استفاده می شوند. ویژگی های رنگ و خط مستقیم با استفاده از تقسیم بندی رنگی با سرعت بالا و روش های استخراج سریع و مستقیم خط مستقیم با استفاده از سیستم های مبتنی بر RKD استخراج می شود. آموزش و الگوریتم های تست طبقه بندی الگو با استفاده از پردازش مبتنی بر RKD است. روش استخراج سریع ویژگی های رنگی از آرایه ای از دستگاه های مبتنی بر RKD استفاده می کند و روش استخراج قطعه قطعه مستقیم و کارآمد با استفاده از آرایه ای از عناصر پردازشی و یک واحد کنترل اصلی است. برخی از دستگاه های همجوشی برای یک خط مستقیم با ویژگی های رنگ استفاده می شوند. منطقه مورد نظر و منطقه بخشهای خط یک شی خاص از دیگر ویژگیهای بهبود دقت طبقه بندی اشیا است.