ISSN: 2350-0328

International Journal of Advanced Research in Science, Engineering and Technology

Vol. 6, Issue 10 , October 2019

Methods for Determining Objects in the Image

Khamdamov Utkir Rakhmatullaevich, Umarov Mukhriddin Abdukhalilugli , Akhmedov Farrukh. Anvarovich

Docent Departments “Hardware and software providing management systems in telecommunications”TUIT named after Muhammad al-Kharazmiy(Uzbekistan, Tashkent)
Assistant Department of Software Engineering Samarkand branch of TUIT(Uzbekistan, Samarkand)
Assistant Department of Software Engineering Samarkand branch of TUIT(Uzbekistan, Samarkand)

ABSTRACT: Some methods for recognizing objects in an image based on border detectors and cascading classifiers are considered.

KEYWORDS: computer vision, OpenCV, boundary detector, convolution matrix, Sobel operator, Canny boundary detector, contour analysis, cascading classifiers, Viola-Jones method, Haar cascade.

I. INTRODUCTION

Computer vision is a technology for creating machines that can detect, track and classify objects. For data processing, it uses statistical methods, as well as models built using geometry, physics, and learning theory. Computer vision is widely used in the management of mobile robots, in surveillance tools, analysis of medical images, as well as in the human-computer interaction interface.

The main section of computer vision is the extraction of information from images or a sequence of images. One of the tasks solved by this section is to determine the object of interest. There are many possible solutions to this problem: the search for circuits, the search for descriptors and special points, the use of neural networks, etc. This article briefly discusses methods for detecting an object based on the search for circuits and the use of cascading classifiers.

II. DEFINING AN OBJECT USING PATH SEARCH

The search for contours is a term denoting a set of mathematical methods aimed at identifying points in a digital image at which the brightness of the image changes dramatically. These points are usually organized as a set of curved lines and are called edges, borders or contours.

Changing the brightness of the image can correspond to: different materials, the difference in the lighting of different parts of the scene, the differences in depth or a change in the orientation of the surface. Ideally, defining the edges helps establish the boundaries and shape of the object.

Selected edges can be of two types: independent and point-dependent. Independent borders display properties such as color and surface shape. Dependents can change at different points of view and display the geometry of the scene.

One of the most used edge selection methods is the Sobel operator. This is a discrete differential operator that calculates the approximate value of the brightness gradient of the image. The result of applying the Sobel operator at each point is the brightness gradient vector or its norm. The Sobel operator is based on the convolution of the image with small integer filters in the vertical and horizontal directions [1].

$$G_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} * A \text{ и } G_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} * A$$

Formula 1. Determination of horizontal and vertical approximate derivatives using Sobel convolution matrices, where Gx and Gy are images containing approximate derivatives, A is the original image, and * is a two-dimensional convolution operation.

At each point in the image, the approximate value of the gradient can be calculated by using the obtained approximate values of the derivatives in the formula-2 by the element-by-element [1].

$$G_i = \sqrt{G_{yi}^2 + G_{xi}^2}$$

ISSN: 2350-0328

International Journal of Advanced Research in Science, Engineering and Technology

Vol. 6, Issue 10 , October 2019

Formula 2. Determining the gradient value of the i-th image element, where G_i is the image element containing the gradient values, and G_{xi} and G_{yi} are image elements containing approximate derivatives. The direction of the gradient vector is calculated by formula-3, which is also used elementwise [1].

$$\theta_i = \arctan\left(\frac{G_{yi}}{G_{xi}}\right)$$

Formula-3. Determination of the gradient value of the i-th image element, where θ_i is the image element containing the direction of the gradient vector, and G_{xi} and G_{yi} are image elements containing approximate derivatives. A big plus of this operator is its simplicity. But the gradient approximation he calculates is very rough. The problem is that the derivative of the differentiable brightness function at any point is the brightness function of all points in the image, and the Sobel operator uses only small neighborhoods of each pixel. Also, due to the small size of the filter, the Sobel operator is very sensitive to noise in the image. Therefore, it represents a very inaccurate approximation of the gradient, but it is of sufficient quality for practical application in many tasks.

In addition to the Sobel operator, other matrix boundary detectors can also be used, such as the Pruitt operator, the Sharr operator, and the Roberts cross operator. These methods are similar to the Sobel operator, but use other convolution matrices [1].

$$G_x = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} * A \text{ and } G_y = \begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix} * A$$

Formula 4. Determination of horizontal and vertical approximate derivatives using Pruitt convolution matrices, where Gx and Gy are images containing approximate derivatives, A is the original image, and * is a two-dimensional convolution operation.

$$G_x = \begin{bmatrix} -3 & 0 & 3 \\ -10 & 0 & 10 \\ -3 & 0 & 3 \end{bmatrix} * A \text{ и } G_y = \begin{bmatrix} -1 & -10 & -3 \\ 0 & 0 & 0 \\ 3 & 10 & 3 \end{bmatrix} * A$$

Formula 5. Determination of horizontal and vertical approximate derivatives using Sharr convolution matrices, where G_x and G_y are images containing approximate derivatives, A is the original image, and * is a two-dimensional convolution operation.

$$G_x = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} * A \text{ and } G_y = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} * A$$

Formula 6. Determination of horizontal and vertical approximate derivatives using Roberts convolution matrices, where

Gx and Gy are images containing approximate derivatives, A is the original image, and * is a two-dimensional convolution operation.

Listing 1.Using matrix border detectors in OpenCV3.x using the Sobel matrix as an example.

```
Mat contour_x, contour_y, kernelMat;
// Kernel definition
float kernel[KERNEL_SIZE][KERNEL_SIZE] = {
                        {-1.0f, -2.0f, -1.0f },
{ 0.0f, 0.0f, 0.0f },
{ 1.0f, 2.0f, 1.0f }     };
// Image upload
 Mat img = imread("muhriddin.jpg");
cvtColor(img, img, CV_BGR2GRAY);
kernelMat = Mat(KERNEL_SIZE, KERNEL_SIZE, CV_32F);
// Initialization of the kernel to search for the horizontal horizontal derivative
for (inti = 0; i<KERNEL_SIZE; i++)  {
for (int j = 0; j <KERNEL_SIZE; j++) {
kernelMat.at(j, i) = kernel[i][j];
        }
}
// Applying a horizontal Sobel matrix to an image
filter2D(img, contour_x, -1, kernelMat);
 // Initialization of the kernel to search for the approximate derivative vertically
for (inti = 0; i<KERNEL_SIZE; i++) {
```

ISSN: 2350-0328
International Journal of Advanced Research in Science, Engineering and Technology
Vol. 6, Issue 10 , October 2019

```
for (int j = 0; j <KERNEL_SIZE; j++) {
kernelMat.at(j, i) = kernel[j][i];
        }
 }
// Applying a vertical Sobel matrix to an image
filter2D(img, contour_y, -1, kernelMat);
// Defining Gradient Values
for (inti = 0; i<img.size().width; i++) {
for (int j = 0; j <img.size().height; j++) {
img.at(j, i) = sqrt(pow(contour_x.at(j, i), 2) + pow(contour_y.at(j, i), 2));
}  }
```

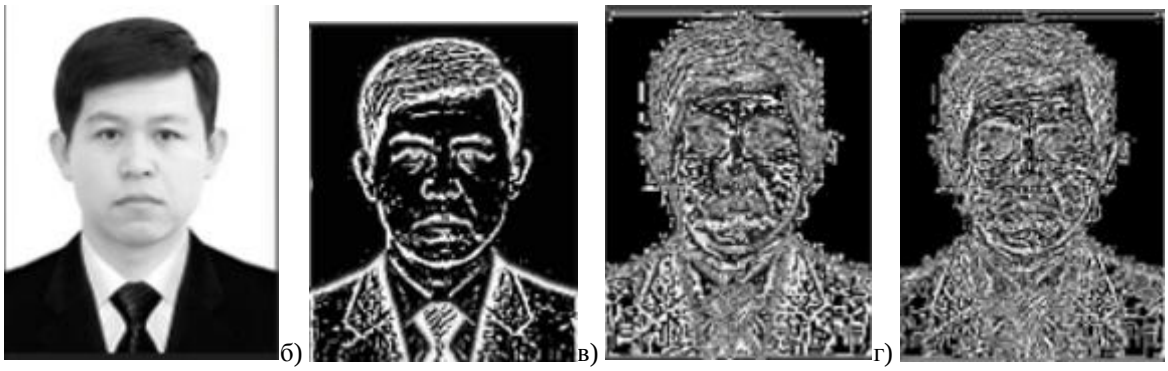

Fig-1. Comparison of the use of convolution matrices: a - original, b - Sharr operator, c - Sobel operator, d - Prewitt operator.

A better and more accurate method is the Canny detector. In 1986, John Canney developed a boundary detector that is optimal for three criteria: low error rate, correct localization, and minimization of responses to one boundary.
In a more expanded sense, this means that the detector should not detect false boundaries (for example, noise), it should correctly and not fragmentarily determine the boundary line, and only react once to each boundary to avoid the appearance of wide bands.
Canny's detector algorithm consists of 5 steps. The first step is smoothing. It is used when, in order to avoid the appearance of false borders, it is necessary to reduce the amount of noise in the image. For this, blurring is often used by a Gaussian filter or some kind of matrix blurring filter [2].

The next two steps are finding the gradients and suppressing the non-maxima. To begin with, all the brightness gradients are found, for this you can use, for example, the Sobel operator described above, but in order for the border to be clear and understandable, it must be represented by a thin line. Therefore, the boundary will be those pixels in which the local maximum of the gradient is reached in the direction of the gradient vector. Assume that almost all the pixels in the gradient are oriented upwards. Then the gradient value in them will be compared with lower and higher pixels. Those pixels that have the greatest value will remain in the resulting image, the rest will be suppressed [2].
And the last stages are double threshold filtering and tracing of the ambiguity region. At this step, one more filtering of false boundaries is performed.

The Canny Boundary Detector uses two thresholds: lower and upper. A pixel whose value is above the upper limit takes the maximum value, i.e., the contour is considered reliable. If the pixel value does not reach the lower threshold, the pixel is suppressed. If its value falls into the range between the thresholds, then it takes an average value, and the decision on whether it is a boundary point will be made during the tracing of the ambiguity region [2].
The trace task is reduced to the distribution of pixels that have received the average value. If such a pixel is in contact with a reliable contour, then its value is equal to the maximum value and it becomes part of the border, otherwise it is suppressed.

OpenCV3.x and higher has a built-in function for filtering Canny - Canny (Mat src, Mat dst, intlowThreshold, inthighThreshold, intkernelSize). Where src is the input black and white image, dst is the output binarized image with the boundaries found, lowThreshold and highThreshold are the lower and upper thresholds, kernelSize is the size of the Sobel matrix.

ISSN: 2350-0328
International Journal of Advanced Research in Science, Engineering and Technology
Vol. 6, Issue 10 , October 2019

Fig-2.Canny Filter Result.

An interesting variety of boundary detectors are angle detectors. There are three categories of angle detectors: extracting information directly from the intensity of the image pixels, methods based on determining the image contour and using models with intensity as parameters.
Defining an object using cascading classifiers

III. CASCADING CLASSIFIERS

Cascading is a special case of ensemble training based on combining several classifiers, using all the information collected from the output of this classifier, as additional information for the next classifier in the cascade. Unlike voting or stacking ensembles, which are multi-expert systems, cascading is multi-stage.

Cascading classifiers are trained with several hundred "positive" samples of individual objects and arbitrary "negative" images of the same size. After training the classifier, it can be applied to the image area and determine the object in question. To search for an object in the entire frame, the search window moves over the entire image with some overlap. This process is most often used in image processing to detect and track objects, primarily for detecting and recognizing faces.
Ensemble methods are a set of weak classifiers (by the weakness of the classifier it is understood that its error in training the sample is less than 50%, but more than 0%). Combining their predictions, it is possible to achieve a higher accuracy of the classification of objects from the test sample [3].

IV. VIOLA JONES METHOD

The Viola-Jones method is an algorithm that allows you to detect objects in an image. The algorithm can recognize various classes of images, but its main task is face detection. In the OpenCV library, it is implemented by the function (cvHaarDetectObjects ()) [4].
A person can easily cope with the task of detecting faces in an image, but the computer needs clear instructions and restrictions. To make the task more solvable, the Viola-Jones method requires a full view of the frontal vertical faces. Thus, in order to be detected, the whole face should look at the camera and should not be tilted in any direction. Although it seems that these restrictions may slightly reduce the usefulness of the algorithm, since the detection step is most often accompanied by a recognition step, in practice these restrictions on the pose are quite acceptable.
Basic principles:

- The use of images in an integral representation. (the integral representation of the image is a matrix whose dimension coincides with the size of the original image), which allows you to quickly calculate the necessary objects. Elements of the matrix can be calculated by the formula.

$$L(x,y) = \sum_{i=0,j=0}^{i\leq x,j\ll y} I(i,j)$$

Formula 7. Calculation of the matrix element of the integrated image representation, where I(i,j) - pixel brightness of the source image.

ISSN: 2350-0328

International Journal of Advanced Research in Science, Engineering and Technology

Vol. 6, Issue 10 , October 2019

- Haar signs are used (Haar signs can be defined as the difference of the sums of pixels of two adjacent areas inside the rectangle), with which the object is searched. The main advantage of the signs of Haar, in comparison with other signs is speed. With an integral representation of the image, Haar features are computed in constant time. In the standard Viola-Jones method, rectangular features are used, they are called Haar primitives.
- The use of boosting (boosting is a general method for increasing the performance of learning algorithms) to select the most suitable features for the desired object in a given part of the image. OpenCV uses the AdaBoost machine learning algorithm. Boosting algorithms are designed to train T "weak" classifiers. Individually, these classifiers are usually simple. Each classifier Kt (t = {1, 2, … T}) associated with the weight at, which is used when combining results from all classifiers. Input feature vector: X = (x1, x2, …xi, …, XM), where i - trait number marked with binary labels yi = {-1, +1} (depending on the boosting algorithm used, the range of values may vary) [6]. The final equation for classification can be represented in the form of formula-8 [5]:

$$F(x) = \text{sign}(\sum_{m=1}^{M} \theta_m f_m(x))$$

Formula 8. The equation for the classification, where denotes a weak classifier, and - the corresponding weight. This is an accurate weighting of a combination of weak M. classifiers.

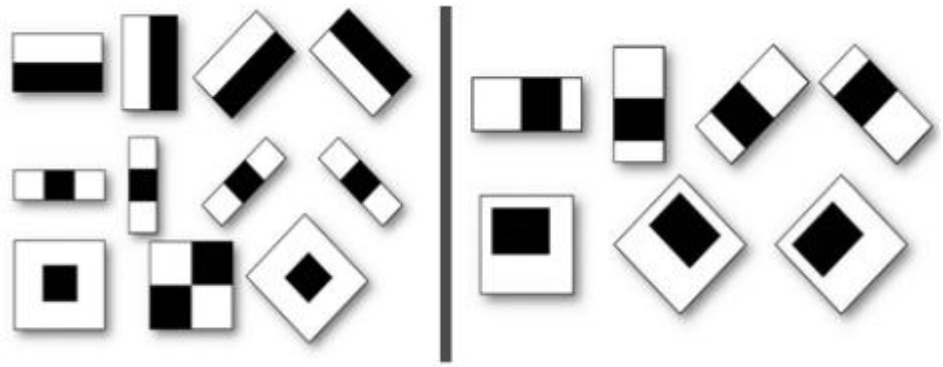

Fig-3.The main and additional signs of Haar.

V. CONCLUSION

Listing 2. Download image and cascades.

```
importnumpy as np
importcv2
face_cascade = cv2.CascadeClassifier('haarcascade_frontalface_default.xml')
eye_cascade = cv2.CascadeClassifier('haarcascade_eye.xml')
img = cv2.imread('muhriddin.jpg')
gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
```

Fig-3.The main and additional signs of Haar.

Listing 2. Download image and cascades.

Listing 3.Defining an object using cascades.

```
faces = face_cascade.detectMultiScale(gray, 1.3, 5)
for (x,y,w,h) in faces:
cv2.rectangle(img,(x,y),(x+w,y+h),(255,0,0),2)
roi_gray = gray[y:y+h, x:x+w]
roi_color = img[y:y+h, x:x+w]
eyes = eye_cascade.detectMultiScale(roi_gray)
for (ex,ey,ew,eh) in eyes:
cv2.rectangle(roi_color,(ex,ey),(ex+ew,ey+eh),(0,255,0),2)
cv2.imshow('img',img)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

ISSN: 2350-0328
International Journal of Advanced Research in Science, Engineering and Technology
Vol. 6, Issue 10 , October 2019

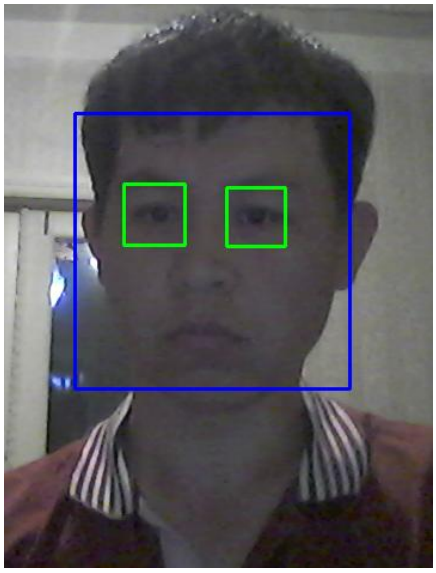

Figure 4.The result of the Haar cascade.

REFERENCES

1. Algorithms for selecting contours of images // Habrahabr.URL: https://habrahabr.ru/post/114452/.
2. Canny Edge Detection // OpenCV.URL: https://docs.opencv.org/3.3.1/da/d22/tutorial_py_canny.html
3. The ensemble method of machine learning, based on the recommendation of classifiers // Free electronic library - electronic materials. URL: http://lib.knigi-x.ru/23raznoe/212318–1-v-state-daetsya-kratkoe-vvedenie-ansambli-klassifikatorov-mashinnom-obuchenii-opisivaet.php.
4. Viola-Jones Method // Wikipedia. URL: https://ru.wikipedia.org/wiki/ Viola-Jones Method.
5. Integral representation of images // Studfiles. URL: https: //studfiles.net/preview/6234768/page: 3 /.