IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received March 30, 2020, accepted April 13, 2020, date of publication April 17, 2020, date of current version May 4, 2020.

*Digital Object Identifier 10.1109/ACCESS.2020.2988496*

# Contour Detection by Simulating the Curvature Cell in the Visual Cortex and Its Application to Object Classification

**ZEKUN CHEN AND RONGTAI CAI**

Fujian Provincial Engineering Technology Research Center of Photoelectric Sensing Application, Fujian Normal University, Fuzhou 350007, China
Key Laboratory of Optoelectronic Science and Technology for Medicine of Ministry of Education, Fujian Normal University, Fuzhou 350007, China
College of Photonic and Electronic Engineering, Fujian Normal University, Fuzhou 350007, China

Corresponding author: Rongtai Cai (gjrtcai@163.com)

This work was supported in part by the Special Funds of the Central Government Guiding Local Science and Technology Development under Grant 2017L3009.

**ABSTRACT** This paper addresses contour detection by simulating the human visual system and its application to visual object classification. Unlike previously designed bioinspired contour detection algorithms, we consider contour to be the salience of an edge image, and we extract the salience by simulating the endstopped cell and curvature cell in the visual cortex. Generally, we follow a local-to-global feed-forward architecture, in which the size of the receptive field (RF) increases from the primary visual cortex to the higher visual cortex. Edges are first detected by simple cells in small RFs, where textural details are suppressed by non-classical receptive fields (NCRFs) and sparse coding. Second, edges are integrated into local segments by complex cells. Afterwards, they are combined into the salience of edge images by endstopped cells and curvature cells and are ultimately the core of the final contour. In addition, we also apply the bioinspired contour detection algorithm to visual object classification tasks. Experiments on contour extraction show that, compared with state-of-the-art bioinspired algorithms, our algorithm makes a considerable improvement on contour detection. Experiments on visual object classification show that the contours produced by our proposal are powerful representations of the original images, which implies that our proposal is both biologically plausible and technologically useful.

**INDEX TERMS** Contour detection, curvature cell, image processing, machine vision, biologically inspired computation, visual object classification.

## I. INTRODUCTION

Contour plays an important part in visual perception, since we can recognize an object just by its silhouette; and it is widely used in plenty of visual tasks, such as visual object detection [1], recognition [2], and tracking [3]. Therefore, contour detection is not only concerned by the biological vision community but also attracts considerable attentions from the machine vision community.

In this paper, we set up a contour extraction system by simulating the visual cortex, and apply it to visual object classification tasks. The motivations of this paper are as follows: (a) Scientifically, the purpose of the proposed work is to provide a possible mechanism of contour perception in biological visual systems. (b) Technologically, one purpose of the proposed work is to provide an unsupervised contour detector, which should be useful in practical visual tasks. (c) The other technological purpose of the paper is to verify the applicability of the proposed contour detector in visual object classification.

The main contributions of this paper are as follows.

First, in contrast to the other biologically inspired contour extraction algorithms [4]–[6], we consider contour to be the salience of an edge image. We extract the salience by simulating the endstopped cell and curvature cell in the intermediate of primary visual cortex and higher visual cortex. Our algorithm is in line with the hierarchical theory of the biological visual system, where contour perception is the bridge linking primary visual perception and higher visual tasks. Our algorithm provides a possible contour sensing method

The associate editor coordinating the review of this manuscript and approving it for publication was Yakoub Bazi.

of the visual cortex, and is proved to be an improvement on bioinspired contour extraction algorithms.

Second, we apply the bioinspired contour algorithm to visual object classification tasks. Though contour extraction by simulating the visual cortex has been studied [4]–[6], it is rarely reported to what extent the contour is applicable to practical visual tasks. There is the work of Rodríguez-Sánchez and Tsotsos [7], but this work is not comprehensive enough. Only a handful of samples are used in [7] to demonstrate the possibility of visual object classification using contour cues. In this paper, we show the applicability of bioinspired contours to visual object classification by experiments on two public image datasets.

The remainder of this paper is organized as follows. In Section II, we report the relevant work. In Section III, we describe our contour detection algorithm by simulating the visual cortex. In Section IV, we briefly show our visual object classification paradigm. In Section V, we conduct the contour detection experiments and object classification experiments based on the bioinspired contour. Finally, we draw the conclusion in Section VI.

## II. RELATED WORK

Contour detection can be traced back to the early work in the 1980s [8]. In those days, the difference of contours and edges was usually neglected, and contour detection was equivalent to edge detection. One purpose of the study of edge/contour detection was aim to reveal how people see the world. For example, Marr and Hildreth proposed a theory of edge detection, and used the theory to explain basic psychophysical findings [8]. Technologically, they proposed a filter for edge detection. Scientifically, they proved that the filter is a physiological model of simple cells in the visual cortex [9], [10].

On the contrary, there are many bioinspired contour detection algorithms based on physiological findings [11]. For example, Wei proposed a contour detection model based on non-classical receptive field [12]. In the model, an image was filtered by retina cells, simple cells and complex cells, then inhibited or disinhibited at different spatial locations on different scales by non-classical receptive field [12]. Meanwhile, Spratling proposed a sparse coding algorithm for boundary detection [5]. Sparse coding is an important property of orientation-tuning cells in the primary visual cortex. Their work was limited to intensity images at a single scale, but outperformed several contour extraction algorithms [5]. On the other hand, Yang and his colleagues proposed an algorithm by simulating the color-opponent mechanisms involved in from the retina to the primary visual cortex for boundary detection in complex natural scenes. Unlike the other previous work, they used the red-green and blue-yellow color opponent channels in the human visual system for boundary detection [4]. Yang and his colleagues then improved their work by mimicking color-sensitive double-opponent cells in the primary visual cortex of the human visual system. The receptive fields used in the algorithm are orientation-sensitive and both chromatically and spatially opponent, In the extended work, they not only extracted boundaries by mimicking the double-opponent cells, but also used the spatial sparseness constraint of neural responses to suppress the textural edges [6]. Another work of Yang and his colleagues is introducing multifeature-based surround inhibitions into contour extraction [13]. They assigned weights to different visual features, like orientation, luminance, and luminance contrast; then they used the weights to modulate the final surround inhibition of the neurons. They improved the performance of contour detection by using the multifeature-based surround inhibitions [13]. Similarly, Akbarinia and his colleagues proposed a biologically-inspired edge detection model [14]. They introduced four receptive field surround modulations into the boundary detection. These were the full, far, iso-orientation and orthogonal-orientation modulations. And they further introduced a feedback connection from higher-level visual areas to the lower ones. By doing so, they made a big improvement compared to the other bioinspired algorithms [14].

Certainly, there are contour detection algorithms that do not based on physiological findings. Plenty of contour detection algorithms originate from applied mathematics. The difference of edges and contours is usually neglected in such kind of algorithm. For example, Wang and Shui proposed a noise-robust color edge detector using gradient matrix and an-isotropic Gaussian directional derivative matrix [15]. Zhang and his colleagues proposed a noise-robust image edge detector using automatic an-isotropic Gaussian kernels [16]. Wang and his colleagues proposed a multiscale edge detector based on first-order derivative of an-isotropic Gaussian kernels [17]. Undoubtedly, Canny operator [18] is one of the most classical methods for edge detection in such kind of algorithm. The other kind of contour detectors come from machine learning technology. For example, Piotr and Zitnick proposed an accurate and computationally efficient edge detection model based on structured-forest [19]. Kivinen and Williams proposed an edge prediction model based on deep neural networks [20]. Xie and Tu proposed a holistically-nested edge detection model based on multi-scale, multi-level feature learning [21]. Liu and Cheng proposed an edge detection model based on richer convolutional features learning [22]. Jiangzhong and his colleagues proposed an edge detection model based on bi-directional cascade network [23].

Apart from the contour detection algorithms, there are practical applications of contour to visual tasks. Due to the well-controlled illuminations and clean background in indoor scenario, one can extract clean and complete contours of objects. The application of contours to these visual tasks, like surface defects detection and visual measurement, is quite successful. For example, Besari introduced surface defect characterization in polishing process based on contour dispersion [24]. Jian used contour information in automatic surface defect detection for mobile phone screen glass. They used contour-based registration to generate an template image, and the template image was used to align

the mobile phone screen glass images [25]. On the other hand, since there are textural details and background clutters, one cannot extract clean and complete contours in complex natural scenes. But the application of contour to such visual tasks is also successful. For example, Shotton applied contour-based learning to object detection [1], and proposed a multi-scale categorical object recognition algorithm based on contour fragments [2]. Yilmaz proposed a contour-based object tracker, and applied the tracker to videos produced by mobile cameras [3]. Whereas, there are rare reports on the application of bioinspired contours to practical visual tasks, except for the report of Rodríguez-Sánchez and Tsotsos [7].

## III. CONTOUR EXTRACTION BY SIMULATING THE HUMAN VISUAL SYSTEM

In this section, we introduce our contour extraction algorithm by simulating the human visual system. FIGURE 1 shows the flowchart of the proposed contour detection algorithm.

As shown in FIGURE 1, our detector begins by generating a yellow (Y) color image and a grayscale image from the original image. All the images are used to generate the color opponent images and spatial opponent images. These are R-G, B-Y, and spatial opponent images. These opponent images are then fed into the Gabor filter mimicking the simple cell in the primary visual cortex (V1), which results in edge images. The textural details in edge images are suppressed by non-classical receptive fields (NCRFs) and sparse coding. Then, nearby edges in the same direction are connected by complex cells. Unlike previously proposed bioinspired approaches, we send the edge image into an additional processing block by mimicking the intermediate visual cortex. We model the curvature cell in the visual cortex to extract curve segments from edge images. These curve segments are the salience of the edge images. However, they are usually incomplete in representing the full contour. Hence, we superpose the edge image with textural details suppressed onto the salience of the edge image, which results in the final contour. Details are given below.

### A. PREPROCESSING BY RETINA CELLS

In the retina, there are cone cells and rod cells that are sensitive to color stimuli and luminance stimuli respectively. For the stimuli, we first produce a grayscale image to imitate the luminance stimulus. Then, we produce a yellow image, denoted as Y. According to the color opponency theory [26], we need a yellow image to generate a color opponent image, denoted as B-Y.

Afterwards, we employ the Gaussian filter to smooth textural details in the R, G, B, Y and grayscale images separately; As a side effect, the edge details are also slightly blurred. As a compensation, we apply the differential of two smoothed images to sharpen the edges, which ultimately produces color opponent images and spatial opponent images. These opponent images are the material for edge extraction.

The opponency mechanism employed by retina cells enhances the edges in images. It facilitates edge extraction for simple cells. Hence, it helps to produce real contours, since contours are parts of edges.

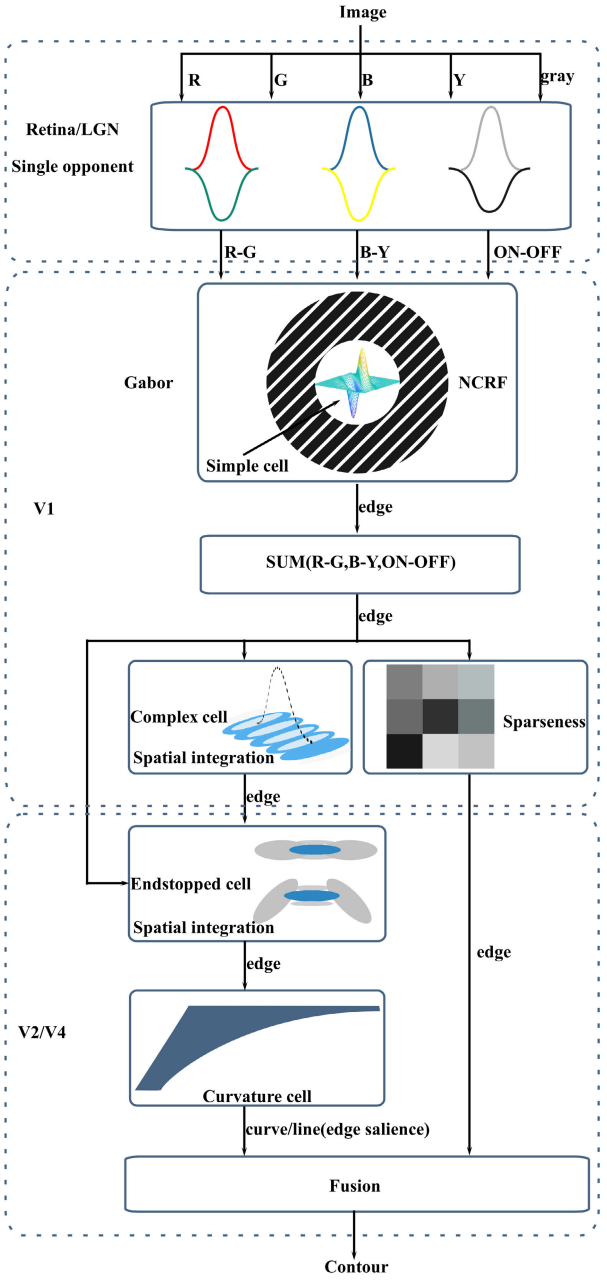



**FIGURE 1.** The flowchart of the proposed contour detector.

### B. EDGE DETECTION BY SIMPLE AND COMPLEX CELLS

Simple cells in the primary visual cortex can be seen as feature detectors with orientation-tuning properties. According to the theory of Hubel and Wiesel [27], simple cells are generated by connecting ganglion cells along the edge orientation. We simplify the architecture of visual cortex by removing the ganglion cells, and replace them with retina cells. Simple cells are usually modeled as 2D Gabor filters [28]. Usually, a maximum pooling step follows the Gabor filter, so that the

responses of simple cells are as follows:

$$R_{step} = \max_{\theta}(R_{retina} * g_{odd,\theta}), \quad (1)$$

$$R_{bar} = \max_{\theta}(R_{retina} * g_{even,\theta}), \quad (2)$$

where * is the convolution operation, $R_{retina}$ is the response of retina cell, $g_{odd,\theta}$ is an odd Gabor filter, $g_{even,\theta}$ is an even Gabor filter, $R_{step}$ is the response of a step-like edge in the direction $\theta$, $R_{bar}$ is the response of a bar-like edge in the direction $\theta$.

Contour is the salience of an edge image, while the textural details in the image are noise that should be removed from the contour image. Therefore, we employ the surround suppression mechanism found in biological vision to suppress the textural details. Physiological evidence shows that the responses of visual neurons are strongly influenced by stimuli presented in the region outside the classical receptive field (CRF) [29]. This region, with a size 2-5 times larger than that of the CRF, is called the nonclassical receptive field (NCRF) [30]. It is defined as:

$$NCRF(\theta) = g_e(\sigma_{x_1}, \theta^{\perp}) - std \cdot g_e(\sigma_{x_2}, \theta^{\perp}), \quad (3)$$

where $\theta^{\perp}$ is the direction of suppression, which is orthogonal to $\theta$, the selective direction of simple cell. $g_e$ is an elliptic Gaussian function along the direction $\theta^{\perp}$, $g_e(\sigma_x, \sigma_y, \theta^{\perp}) = e^{-(Ax^2-2Bxy+Cy^2)}$, $A = \frac{cos^2\theta^{\perp}}{2\sigma_x^2} + \frac{sin^2\theta^{\perp}}{2\sigma_y^2}$, $B = \frac{cos2\theta^{\perp}}{4\sigma_x^2} + \frac{sin2\theta^{\perp}}{4\sigma_y^2}$, $C = \frac{sin\theta^{\perp 2}}{2\sigma_x^2} \cdot \frac{cos\theta^{\perp 2}}{2\sigma_y^2}$. We set $\sigma_{x_1} = 3.65$, $\sigma_{y_1} = \frac{1}{8}\sigma_{x_1}$, $\sigma_{x_2} = 3\sigma_{x_1}$ and $\sigma_{y_2} = \frac{1}{8}\sigma_{x_2}$, according to the electrophysiological measurements in [31]. *std* is the local standard deviation of an image patch. The response of the NCRF, $R_{NCRF}$, is the convolution of $R_{step}/R_{bar}$ and $NCRF(\theta)$.

Furthermore, we apply sparse coding to $R_{NCRF}$. Sparseness is defined as [32]

$$sparseness(x, y; \overrightarrow{h}) = \frac{\left(\sqrt{n} - \frac{\|\overrightarrow{h}(x,y)\|_1}{\|\overrightarrow{h}(x,y)\|_2}\right)}{\sqrt{n}-1}, \quad (4)$$

where $\overrightarrow{h}(x, y)$ is the magnitude histogram of the gradient of a local region centered at $(x, y)$ and $n$ is the dimension of $\overrightarrow{h}(x, y)$, which is set to 100 in our experiments to achieve a balance of computational complexity and performance.

The response of sparsity is the dot product of sparseness and $R_{NCRF}$:

$$R_{sparse} = sparseness(x, y; \overrightarrow{h}) \cdot R_{NCRF}. \quad (5)$$

According to the hierarchical theory of Hubel and Wiesel [33], the responses of simple cells are processed by complex cells. A complex cell is generated by connecting simple cells along $\theta^{\perp}$, orthogonal to the selective direction of the simple cells $\theta$. It is sensitive to orientation tuning as a simple cell is, but it has a larger receptive field. This implies that, similar to a simple cell, a complex cell is an edge detector, but with better robustness. It may play the role of connecting edge segments along a specific direction while allowing small displacements. The definition of a complex cell is

$$R_{CC} = \sum_{i=1}^{n} c_i(R_{SC_i}), \quad (6)$$

where $R_{SC_i}$ is the response of the $i$-th simple cell, $c_i$ is the $i$-th weight. These weights are inversely proportional to the distances of the cells to the center of the RF, and follow a Gaussian distribution. According to the hierarchical theory of Hubel and Wiesel [33], we connect 5 simple cells to a complex cell, and the weights are set to 0.1350, 0.6049, 0.9974, 0.6049 and 0.1350.

Simple and complex cells extract edges from opponent images, which provide material for the next phase of contour detection. The NCRF and sparse coding suppress the textural details in the edge images. Ideally, there should be no textural details in the final contour image, and the precision of contour detection should be largely improved.

### C. SALIENCE DETECTION BY ENDSTOPPED CELLS AND CURVATURE CELLS

As shown in the previous section, complex cells detect larger line segments than simple cells do. Similarly, for the detection of line segments beyond the scope of complex cells, endstopped cells [34] are needed. An endstopped cell is generated by connecting a simple cell and two displaced complex cells [35], whose orientations are identical:

$$R_{EC} = \phi(c_{ctr}\phi(R_{ctr}) - (c_{dp1}\phi(R_{dp1}) + c_{dp2}\phi(R_{dp2}))), \quad (7)$$

where $c_{ctr}$, $c_{dp1}$ and $c_{dp2}$ are the gains of the center cell and the two displaced cells, respectively. According to our experiments, we can achieve good performance simply by setting the gains to 1; that is, $c_{ctr} = c_{dp1} = c_{dp2} = 1$. $R_{ctr}$, $R_{dp1}$ and $R_{dp2}$ are the responses of the center cell and the two displaced cells, respectively. $\phi$ is a rectification function that zeros negative values. Therefore, the response of the endstopped cell $R_{EC}$ is the linear combination of the response of the simple cell $R_{ctr}$ and the responese of the two complex cells $R_{dp1}$ and $R_{dp2}$, but with rectification both before and after combination.

As mentioned, contour extraction is performed in a larger region beyond the RF of a complex cell. In this case, image edges may present in the form of curves. It is not reasonable to extract them by using endstopped cells. Therefore, filters that are suitable for curves, such as curvature cells, are needed. Dobbins proposed a curve detector composed of a central cell and two displaced cells with different orientations [36]. For example, the orientations 0°, 45° and 135° could be used for the central simple cell and the two displaced complex cells, respectively. The responses of a curve detector are:

$$R_u = \phi(c_{ctr}\phi(R_{ctr}) - (c_{dp_{45}}\phi(R_{dp_{45}}) + c_{dp_{135}}\phi(R_{dp_{135}}))), \quad (8)$$

$$R_l = \phi(c_{ctr}\phi(R_{ctr}) - (c_{dp_{135}}\phi(R_{dp_{135}}) + c_{dp_{45}}\phi(R_{dp_{45}}))), \quad (9)$$

where $c_{ctr}$, $c_{dp_{45}}$ and $c_{dp_{135}}$ are the gains of the center cell and the two displaced cells, respectively. According to our experiments, we can achieve good performance simply by setting the gains to 1; that is, $c_{ctr} = c_{dp_{45}} = c_{dp_{135}} = 1$. $R_{ctr}$, $R_{dp_{45}}$ and $R_{dp_{135}}$ are the responses of the central cell and the two displaced cells, respectively. $R_u$ is the curve of the top half of a circle and $R_l$ is the curve of the bottom half of a circle. Similar to the response of endstopped cell $R_{EC}$, the response of the curve detector $R_u/R_l$ is the linear combination of the responses of the simple cell $R_{ctr}$ and the two complex cells $R_{dp_{45}}$ and $R_{dp_{135}}$, but with rectification both before and after combination. $\phi$ is a rectification function that zeros negative values. The difference between the endstopped cell and the curve detector is whether the three cells are in the same direction; see eq (7) and eqs (8-9) for examples.

The response of the curvature cell is a combination of the response of a endstopped cell and the response of a curve detector:

$$R_{\theta,r,s_p} = R_{EC} \cap (R_u > R_l), \tag{10}$$

$$R_{\theta,r,s_n} = R_{EC} \cap (R_u < R_l), \tag{11}$$

where $R_{\theta,r,s}$ is the response of the neuron in the direction $\theta$, with curvature $r$ and sign $s$ ($p$ is positive and $n$ is negative). The expression $R_u > R_l$ produces a mask image. If $R_u(x, y) > R_l(x, y)$, the pixel at $(x, y)$ is set to 1; otherwise, it is set to 0. The response of the curvature cell is the intersection of $R_{EC}$ and the mask image. Eq (11) works in the same manner as eq (10) does.

A maximum pooling step follows each curvature cell:

$$R_{curve,k} = \max_{\theta,r,s}(\theta, r, s, k), \tag{12}$$

which selects the maximum response from the set of all possible directions, curvatures and signs as the curve at location $k$. We call the responses of curvature cells the salience of an edge image. The endstopped cells and curvature cells extract the salience of an edge image, which is ultimately the core of the contour of the image. There are no textural details in the core of the contour. And the core of the contour is enhanced by the method presented here, so that the performance of contour detection is improved.

### D. COMPENSATING THE SALIENCE OF THE EDGE IMAGE

It is insufficient to represent the contour of an image only by using the salience of the edge image, since some edges belonging to the contour are absent from the salience. As compensation, we superpose the edge image with textural details suppressed on the salience of the edge image. Consequently, contour is the superposition of the salience of an edge image and the response of sparse coding:

$$R_{contour} = \alpha_1 R_{curve} + \alpha_2 R_{sparse}, \tag{13}$$

where $R_{contour}$ is the final contour response; $\alpha_1$ and $\alpha_2$ are the weights. According to our experiments, we can achieve good performance simply by setting the weights to 1; that is, $\alpha_1 = \alpha_2 = 1$.

## IV. OBJECT CLASSIFICATION VIA CONTOUR IMAGES

In this paper, we follow the scheme of traditional visual object recognition. Therefore, a feature detector and a classifier are needed. We extract the histogram of gradient orientations (HoGs) from the contour images for classification. The HoG is a powerful descriptor of image features proposed by Dalal and Triggs [37], which is widely used in visual object detection, recognition and tracking. Meanwhile, we employ the popular support vector machine (SVM), also called the support vector classifier (SVC) [38], as the classifier.

## V. EXPERIMENTS AND DISCUSSION

In this section, we report experiments that evaluate the performance of our contour detector. Our experiments are carried out on a PC, with an intel i5-8500 CPU at 3.00 GHz and 8 GB memory, in MATLAB 2017. The parameter values used in the experiments were given along with the corresponding equations in section III. And the code of our contour detector is available at https://github.com/zekunchen/Contour-Detection.

This section consists of contour extraction experiments, image classification experiments based on the contours and discussion based on the experiments. We first compare the performance of our contour detection algorithm with the state-of-the-art contour detectors in section A. Then we show the performance difference of visual object classification between our proposals and the state-of-the-art algorithms in sections B and C. We also compare the classification performance by using contour images and those by using the original images in sections B and C. Comprehensive discussion based on the experiments is presented in section D.

### A. EXPERIMENTS ON CONTOUR EXTRACTION BY SIMULATING THE VISUAL CORTEX

#### 1) EXPERIMENT DATASETS

In these experiments, we compare our contour detector with eight state-of-the-art contour detectors by testing them on the Berkeley Segmentation Dataset and Benchmark (BSDS) [39]. The BSDS dataset is widely used for the research on image segmentation and boundary detection. Aside from the original images, the dataset also provides hand-labeled segmentations of dataset images from 30 human subjects. The dataset includes a color image version of BSDS300, a grayscale image version of BSDS300, a color image version of BSDS500, and a grayscale image version of BSDS500.

Since there are no training phases in the bioinspired algorithms, all of the images are used for testing. Whereas, for the machine-learning based algorithms, the images in the BSDS300 are divided into a training set of 200 images and a test set of 100 images; while the images in the BSDS500 are divided into a training set of 300 images, and a test set of 200 images.

### 2) COMPARISON ALGORITHMS

To examine the performance of our contour detection algorithm, we compare our contour detection algorithm with eight state-of-the-art algorithms. These algorithms are CO [4], MCI [13], SCO [6], gPb [40], DeepNets [20], HED [21], RCF [22] and BDCN [23]. The CO [4], MCI [13] and SCO [6] contour detectors are biologically motivated; The gPb contour detector [40] is a traditional machine learning algorithm. The DeepNets [20], HED [21], RCF [22] and BDCN [23] detectors are based on deep neural networks (DNNs). For the bioinspired contour detectors and the gPb detector [40], we used the publicly available source code in the experiments, we ran the programs with the default values of the parameters provided by the authors. For the DNN-based contour detectors, we used the results as they are reported for comparison.

### 3) EVALUATION METRICS

For a quantitative performance comparison, we report three evaluation criteria in the experiments. These are the best F-measure on the dataset for a fixed scale (ODS), the aggregate F-measure on the dataset for the best scale in each image (OIS), and the average precision (AP) on the full recall range [40]. AP is equal to taking the area under the precision-recall (PR) curve. While, the F-measure is defined as

$$F = \frac{(\alpha^2 + 1)Precision \cdot Recall}{\alpha(Precision + Recall)}, \tag{14}$$

where $precision = \frac{TP}{TP+FP}$, $recall = \frac{TP}{TP+FN}$, $TP$ is the number of true positives, $FP$ is the number of false positives, and $FN$ is the number of false negatives. We set $\alpha = 1$, so that $F = F_1 = \frac{2 \cdot Precision \cdot Recall}{Precision + Recall}$.

Apart from the three criteria ODS, OIS and AP, F-measure and PR curve are also used for the evaluations in the experiments.

**TABLE 1. Results of several contour detectors on BSDS300 and BSDS500 (Color Image Version).**

| Method | BSDS300 | | | BSDS500 | | |
|---|---|---|---|---|---|---|
| | ODS | OIS | AP | ODS | OIS | AP |
| Human | 0.79 | 0.79 | - | 0.80 | 0.80 | - |
| Biological | | | | | | |
| CO[4] | 0.64 | 0.66 | 0.65 | 0.64 | 0.68 | 0.65 |
| MCI[13] | 0.62 | - | - | 0.64 | - | - |
| SCO[6] | 0.66 | 0.68 | 0.70 | 0.67 | 0.71 | 0.71 |
| **Ours** | **0.68** | **0.70** | **0.70** | **0.70** | **0.72** | **0.72** |
| Machine-learning | | | | | | |
| gPb[40] | 0.70 | 0.72 | 0.66 | 0.71 | 0.74 | 0.65 |
| Deep-learning | | | | | | |
| DeepNets[20] | - | - | - | 0.74 | 0.76 | 0.76 |
| HED[21] | - | - | - | 0.78 | 0.80 | 0.83 |
| RCF[22] | - | - | - | 0.81 | 0.82 | - |
| BDCN[23] | - | - | - | 0.83 | 0.84 | 0.90 |

### 4) EXPERIMENT RESULTS

TABLE 1 reports the quantities of ODS, OIS and AP of different contour detectors for comparison.

It is obvious from TABLE 1 that our detector makes a considerable improvement compared with the other bioinspired methods. For example, compared with SCO [6], the improvement of ODS for our detector is more than 4% on the BSDS500 and 3% on the BSDS300. However, our detector underperforms DNN-based detectors such as DeepNets [20], HED [21], RCF [22] and BDCN [23]. None of the bioinspired detectors performs as good as the DNN-based detectors.

In addition, we also compare our contour detector with the other contour detectors on the grayscale image version of BSDS, as shown in TABLE 2.

**TABLE 2. Results of several contour detectors on BSDS300 and BSDS500 (Grayscale Image Version).**

| Method | BSDS300 | | | BSDS500 | | |
|---|---|---|---|---|---|---|
| | ODS | OIS | AP | ODS | OIS | AP |
| Human | 0.79 | 0.79 | - | 0.80 | 0.80 | - |
| Biological | | | | | | |
| CO[4] | 0.60 | 0.63 | 0.60 | 0.61 | 0.64 | 0.61 |
| SCO[6] | 0.62 | 0.64 | 0.64 | 0.63 | 0.67 | 0.66 |
| **Ours** | **0.64** | **0.66** | **0.65** | **0.65** | **0.68** | **0.66** |
| Machine-learning | | | | | | |
| gPb[40] | 0.61 | 0.64 | 0.60 | 0.63 | 0.66 | 0.62 |

For the ODS criterion, our detector has an improvement of 3% both on BSDS500 and on BSDS300 over SCO [6]. Note that the performance of the machine-learning-based gPb detector [40] on grayscale images is significantly degraded, whereas our detector outperforms gPb by 4% and 3% on BSDS300 and BSDS500, respectively. To the best of our knowledge, there are no reports of DNN-based contour detection on the grayscale image version of BSDS300 and grayscale image version of BSDS500.

According to the results, we conclude that bioinspired methods are more robust than learning-based detectors such as gPb [40], DeepNets [20], HED [21], RCF [22], and BDCN [23], since bioinspired methods are independent of the dataset, while learning-based methods are dependent on the dataset. If applied to other datasets, the performance of learning-based detectors may degrade greatly, while the bioinspired methods may perform similarly.

Considering the fact that DNN-based detectors use ground-truth images for training, but bioinspired detectors work in an unsupervised manner. It is more reasonable to compare our detector with the other bioinspired detectors. So we focus on the comparison of our detector with the other bioinspired methods. The PR curves for color image version of BSDS300 and BSDS500 are shown in FIGURE 2. It is obvious from FIGURE 2 that the proposed contour detector outperforms the other bioinspired detectors on both BSDS300 and BSDS500.

FIGURE 3 demonstrates a qualitative comparison of different contour detection algorithms. These are the original image, the ground-truth contour image, and the contour images produced by CO [4], SCO [6], HED [21] and our detector.

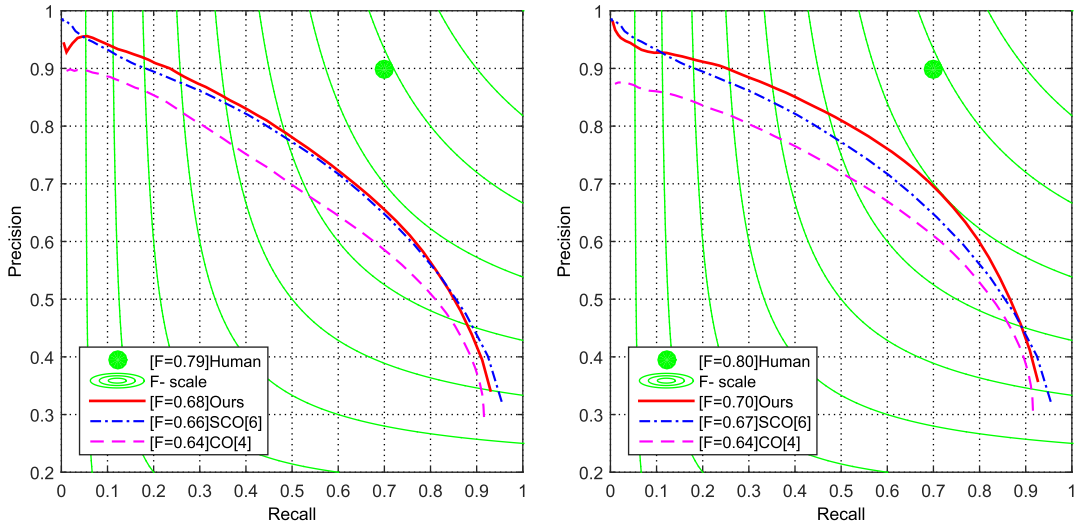



**FIGURE 2.** PR curves of color images from BSDS300 (left) and BSD500 (right).

Compared with CO [4], our contour detector enhances the contour details while suppresses the textural details. Whereas CO enhances both the contour segments and the textural details. Hence, our detector obtains a higher score than CO. Examples are presented in the third column and the last column of FIGURE 3. See the image regions bounded by blue rectangles in the images for details. Compared with SCO [6], our detector enhances the contour segments wile suppresses the texture details. Whereas the SCO suppresses both the contour segments and the textural details. Therefore, our detector has a higher score than SCO. Examples are given in the fourth column and the last column of FIGURE 3. See the image regions bounded by red ovals for comparison.

According to the experimental results, we conclude that our detector can split an edge image into a contour part and a textural-detail part to a certain extent. We achieve this goal by the employment of the endstopped cells and curvature cells. Most of the cells are found in area 4 of the visual cortex (V4), where the shape of an object is encoded [11]. Image contour is the intermediary between the low-level edge features and high-level representation of an object (shape). We believe that the employment of the endstopped cells and curvature cells is the right way to obtain image contour. However, CO and SCO cannot split an edge image into a contour part and a textural-detail part because CO and SCO only employ the simple and complex cells, which are only associated with low-level edge encoding.

Regarding HED [21], a DNN-based contour detection algorithm, our detector underperform HED in extracting contours, and HED obtains a higher score than our detector. The reason is that HED uses the ground-truth contour images for training, but our detector does not employ any ground-truth information. Subjectively, it is doubtful whether the contour produced by HED is better than that of our detector. Examples are given in the last two columns of FIGURE 3. See the regions bounded by blue rectangles in the images for details.

The F-Measure for an individual image is presented in the bottom right corner of each contour image in FIGURE 3. It is clear from FIGURE 3 that our detector outperforms the bioinspired CO detector and the SCO detector both qualitatively and quantitatively.

### 5) RUNNING TIMES

As well as the results showed above, we also report the running times of six algorithms on the color image version of BSDS300 and BSDS500, as shown in TABLE 3.

**TABLE 3.** Average calculation time (in s) of several edge detectors on BSDS300 and BSDS500 dataset.

| Method | Time (s) | Computer configuration |
|---|---|---|
| Canny[18] | $\approx 0.54$ | Intel(R) Core(TM) i5-8500 CPU @ 3.00GHz |
| CO[4] | $\approx 0.78$ | |
| MCI[13] | $\approx 12$ | |
| SCO[6] | $\approx 1$ | |
| gPb[40] | $\approx 110$ | |
| **Ours** | $\approx$ **11*** | |
| DeepNets[20] | $\approx 0.2$ | GPU |
| HED[21] | $\approx 1$ | |
| RCF[22] | $\approx 0.125$ | |
| BDCN[23] | $\approx 0.045$ | |

*Matlab implementation of our algorithm

In the experiments, the average running time of our contour detector using MATLAB is about 11 s per image. Our detector runs slower than the other algorithms. The reason is that the ''imfilter'' function in MATLAB is very slow, and it is widely employed in our program. A C++ implementation of our program with the ''filter2D'' function in OpenCV will run much faster. Empirically, the program will run approximately 10 times faster. The DNNs run faster than the bioinspired methods due to the employment of computers

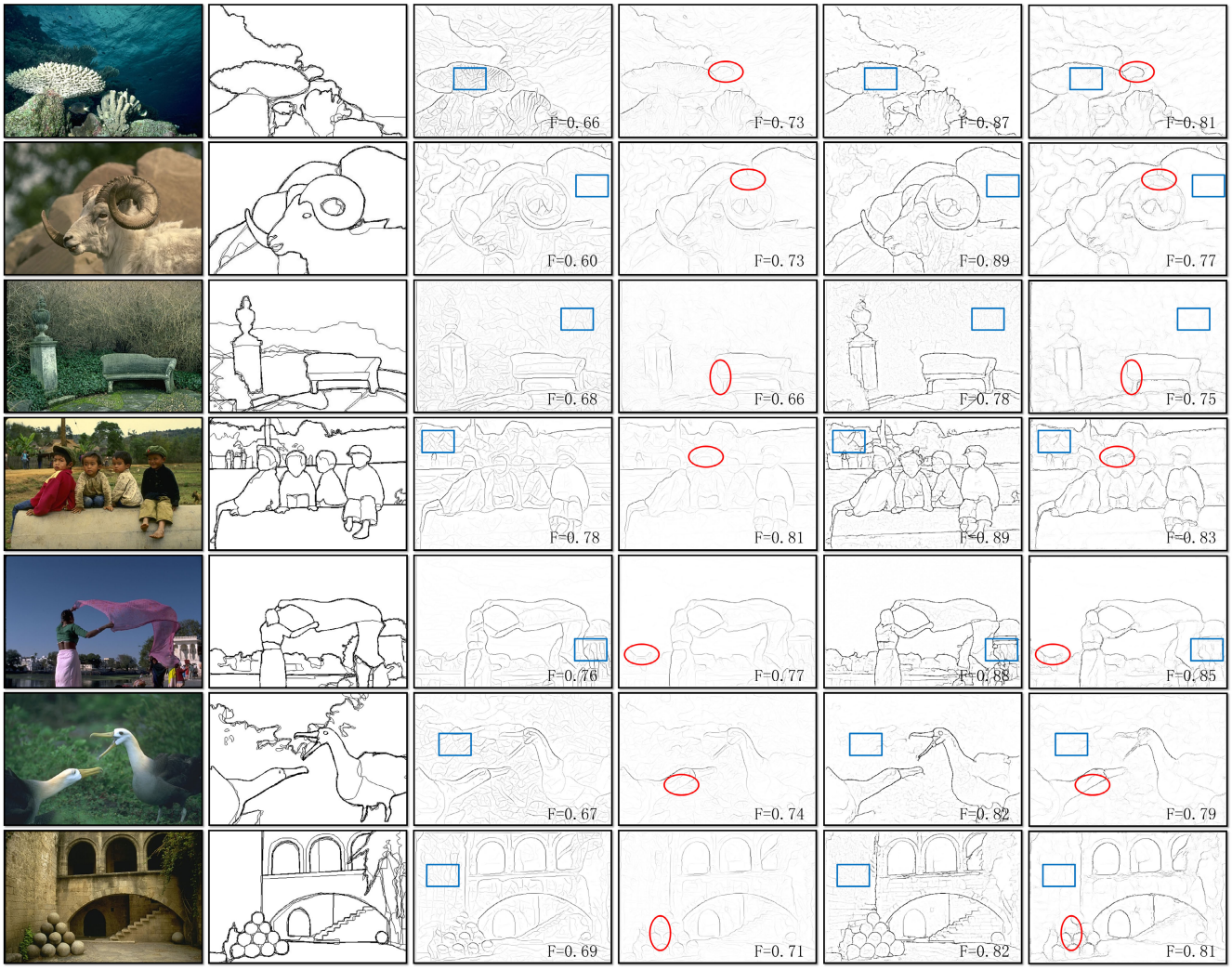



**FIGURE 3.** Subjective comparison of our proposal with several contour detectors.

with high-performance GPUs. However, we ran the bioinspired detectors on an average computer without GPUs. Here, we show the computation times of DNNs as they are reported by the authors. Furthermore, considering the feed-forward architecture of our detector, hardware implementation of our detector is very easy. An FPGA (Field Programmable Gate Array) version of our detector will run much faster, since only hardware delays contribute to the computation time.

### B. CLASSIFICATION EXPERIMENTS ON FASHION MNIST

First, to verify the applicability of our contour detection algorithm, we apply it to classification experiments, and compare the classification performance with different algorithms. Then, to evaluate the power of the contours produced by our proposal as representations of visual objects, we compare the classification performances by using the contour images vs by those by using the original images. The Fashion MNIST dataset [41] and the Swedish Leaves dataset [42] are used in these experiments. This section discusses the experiment on the Fashion MNIST dataset. The experiment on the Swedish Leaves dataset is discussed in the next section.

#### 1) EXPERIMENT DATASETS

The Fashion MNIST dataset is designed to replace the MNIST dataset for bench-marking machine learning algorithms. There are 60,000 images in the training set, 10,000 images in the test set. These images distribute in 10 categories such as T-shirt, trousers, pullover, dress, coat, sandal, shirt, sneaker, bag and ankle boot. The original image are grayscale, each one of which has a size of $28 \times 28$. We resize each one to be the size of $64 \times 64$. FIGURE 4 demonstrates ten examples of the images and their contour counterparts produced by our proposal.

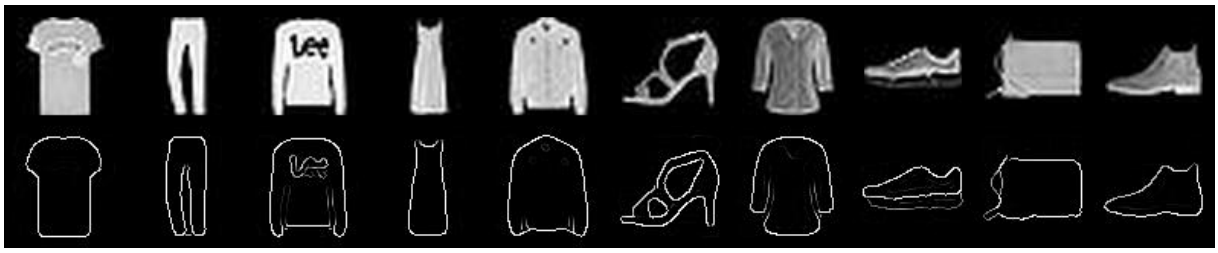


**FIGURE 4.** Image samples and their contours in the Fashion MNIST dataset.

It is obvious that the contours are quite clean and complete. We find that the textural details of different samples in a category can be very different; therefore, they are not suitable for classification. Additionally, there are no other valid features, such as color, for classification. Hence, only image contours are available for classification. We expect that contours are powerful representations of the original images for classification.

### 2) COMPARISON ALGORITHMS

We compare our proposals with nine different algorithms to evaluate the performance of our proposal. One part of the comparison algorithms are reported in the Benchmark [43], including "ExtraTreeClassifier," "DecisionTreeClassifier," "LogisticRegression" , "KNeighborsClassifier," "RandomForestClassifier," "GradientBoostingClassifier" and SVC. The other part of the comparison algorithms are based on deep neural networks. These are "2conv + pooling" and ResNet18 [44].

Besides, we also compare the classification performance by using the original images vs by using the contour images produced by our contour detector to assess the power of the contours as the representations of the original images.

### 3) EVALUATION METRICS

First, we use confusion matrix to visually show where and how the our proposal is wrong and where and how it is correct in classification. Then, we use the accuracy of classification to evaluate the performance of the algorithms. The accuracy of classification is the ratio of the sum of true positives and true negatives to total cases, it is defined as

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}, \tag{15}$$

where $TP$ the number of the true positives, $TN$ is the number of true negatives, $FP$ is the number of false positives, and $FN$ is the number of false negatives.

### 4) EVALUATION METRICS

To verify the performance of the produced contours in representing the objects, we assess the classification performance by using the HoG features extracted from the contours vs those extracted from the original images. Both the original images and the contour images are reshaped into vectors by means of HoGs. The classification results by using contour images are shown in the confusion matrix in FIGURE 5.

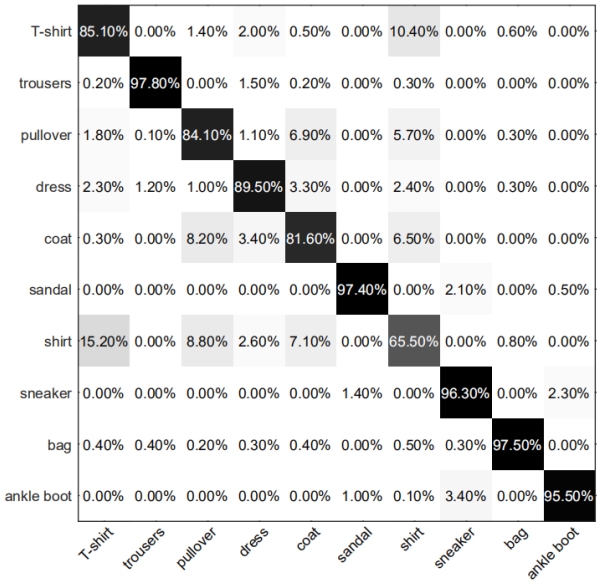



**FIGURE 5.** Classification confusion matrix of the Fashion MNIST dataset using contour images and HoGs.

The classification accuracy varies from class to class, ranging from 65.5% to 97.8%. This implies that the contours of objects along with HoG descriptors are sufficient for classification for some categories but not sufficient for classification for the other categories. The result by using the original images is almost the same as that by using the contour images, so we do not show it for conciseness.

We compare our proposals with nine different algorithms according to the classification accuracy, as showed in TABLE 4.

In TABLE 4, we select the best results from the benchmark [43] for comparison. Most of the algorithms outperform human beings. In the traditional algorithms, the algorithms based on SVC generate the best results. Hence, we use the SVC (SVM) as the classifier in our experiments.

**TABLE 4. Classification results of the Fashion MNIST dataset.**

| Publication | Image+Feature | Classifier | Accuracy |
|---|---|---|---|
| Benchmark[43] | Original | Human | 83.50% |
| | Original | ExtraTreeClassifier | 77.90% |
| | Original | DecisionTreeClassifier | 80.10% |
| | Original | LogisticRegression | 84.20% |
| | Original | KNeighborsClassifier | 86.00% |
| | Original | RandomForestClassifier | 87.90% |
| | Original | GradientBoostingClassifier | 88.80% |
| | Original | SVC | 89.70% |
| [44] | Original | 2conv+pooling | 91.60% |
| | Original | ResNet18 | 94.90% |
| **Ours** | Original+HoG | SVM | 90.17% |
| | Contour+HoG | SVM | 89.03% |

The results are quite encouraging compared with the benchmarks, although our algorithms underperform DNN-based classifiers ''2conv + pooling'' and ResNet18 [44].

Importantly, the accuracy by using contour images is only slightly lower than the accuracy by using the original images, as shown in the last two rows of TABLE 4. It implies that the contour images are satisfactory representations of the original images, and the proposed contour extractor is useful in practical visual tasks.

### C. CLASSIFICATION EXPERIMENTS ON SWEDISH LEAVES

To evaluate the applicability of our contour detector and the power of the bioinspired contours as the representations of original images, we conduct another experiment on the Swedish Leaves dataset [42].

#### 1) EXPERIMENT DATASETS

The Swedish leaves dataset contains 1125 leaves. These leaves are from 15 species, including Ulmus carpinifolia, Acer, Salix aurita, Quercus, Alnus incana, Betula pubescens, Salix alba Sericea, Populus tremula, Ulmus glabra, Sorbus aucuparia, Salix sinerea, Populus, Tilia, Sorbus intermedia and Fagus silvatica. There are 75 leaves per class. In this experiment, 50 leaves per class are used for training, 25 leaves per class are used for test. FIGURE 6 shows 15 leaves along with their contour counterparts produced by our proposal.

The images in this dataset have high resolutions than those in the Fashion-MNIST dataset. Therefore, Our proposal produces much more complete contours. In addition, there are no strong textural details in the contour images. Therefore, the contour images are much cleaner than those produced from the Fashion-MNIST dataset.

#### 2) COMPARISON ALGORITHMS

To evaluate the performance of our proposal in classification, we compare our proposal with eight different paradigms. These are ''color moments + gray level + Naive Bayes classifier [45],'' ''Co-ccurrence + gray level + Random forest classifier [45],'' ''Run Length + LBP-GWO + J48 classifier [45],'' ''PSO-segmentation + LBP-GWO + SVM [46],'' ''ZM + Hog + SVM [47],'' ''CNN [48]'' and ''CNN + SVM [48]''.

Besides, to check the power of the contours as the representations of the original images, we also compare the classification performance by using the original images vs by using the contour images produced by our contour detector.

#### 3) EVALUATION METRICS

The evaluation metrics used in this section are exactly the same as those used in section B. First, we use confusion matrix to visually show where and how the our proposal is actually wrong and where and how it is correct in classification. Then, we use the classification accuracy defined by Eq(15) to evaluate the performance of the algorithms, and compare it with seven different paradigms.

#### 4) EXPERIMENT RESULTS

Similar to the classification experiments on the Fashion MNIST dataset, the original images and the contour images are reshaped into vectors for classification by means of HoGs. The confusion matrix in FIGURE 7 demonstrates the experimental results by using the contour images.

The results are much better than those of the Fashion MNIST dataset. The classification accuracy varies from class to class, ranging from 96% to 100%. This implies that the contours of objects along with HoG descriptors are sufficient for the classification. The experimental setting is identical to that in the experiment on the Fashion MNIST dataset. But high resolution images result in high quality contours, which in turn result in high classification accuracy. The results by using the original images are almost the same as those by using the contour images, therefore we do not show them again.

Beside, we compare our classifiers with seven available algorithms according to the classification accuracy, as shown in TABLE 5.

**TABLE 5. Classification results of the Swedish Leaves dataset.**

| Publication | Image+Feature | Classifier | Accuracy |
|---|---|---|---|
| [45] | Color moments + Gray Level | Naive Bayes classifier | 83.50% |
| | Co-occurrence +Gray Level | Random forest classifier | 77.90% |
| | Run Length | J48 classifier | 80.10% |
| [46] | PSO-segmentation + LBP-GWO | SVM | 93.30% |
| [47] | ZM + HoG | SVM | 98.13% |
| [48] | CNN | - | 98.23% |
| | CNN | SVM | 99.12% |
| **Ours** | Original+HoG | SVM | 98.67% |
| | Contour+HoG | SVM | 98.67% |

In TABLE 5, we use five traditional algorithms and two DNN-based algorithms for the comparison. In the traditional algorithms, the HoG feature outperforms the other image features such as color moments, Zernik moments (ZM), and local binary patterns (LBPs). Hence, we speculate that the HoG feature is the most effective non-learning feature for leaf species classification, and we involve HoGs into our classification paradigms. Compared with the other non-learning algorithms, our algorithms produce better results. Our algorithms also outperform CNN [48] in this experiment, while

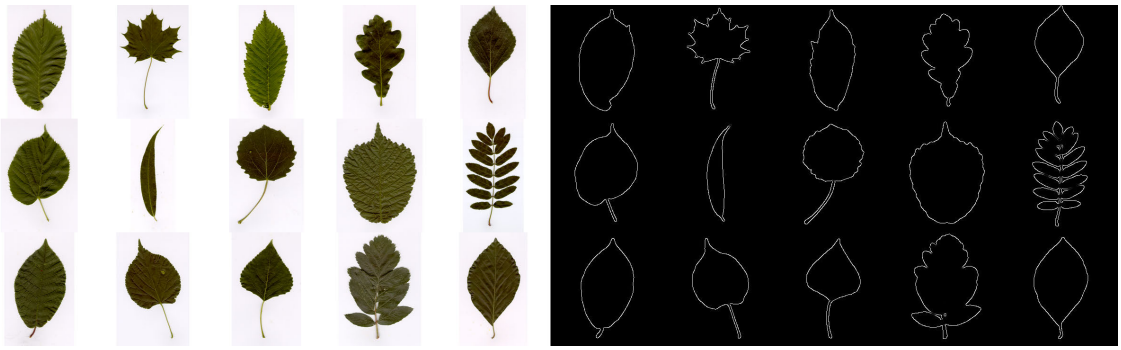


**FIGURE 6.** Image samples and their contours in the Swedish Leaves dataset.

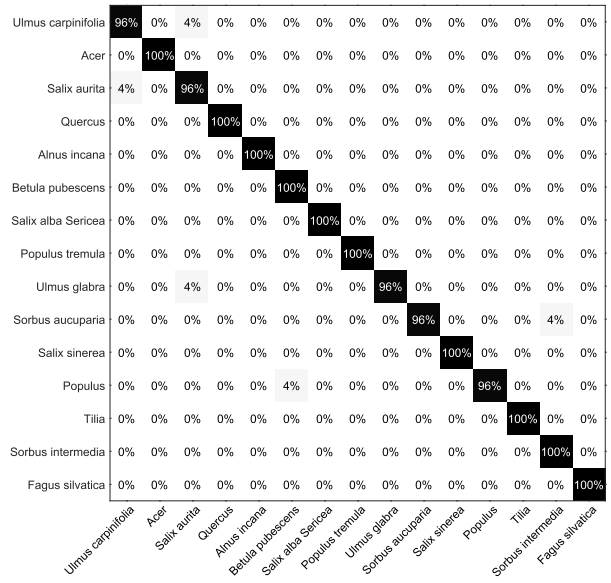



**FIGURE 7.** The confusion matrix of the classification experiment on Swedish Leaves dataset using contour images and HoGs.

underperform the paradigm of ''CNN + SVM'' [48], but the difference is very small.

Importantly, the result by using the contour image is exactly the same as the result by using the original image, as showed in the last two rows in TABLE 5. This implies that the contour produced by our proposal is the apt representation of the original image for classification. Because the contour image contains only contour pixels that are assigned with nonzero values, they are the sparse coding of the original images, and they seem to be the representations of objects in the intermediate visual cortex. The results prove that the proposed contour extractor is both biologically plausible and technologically applicable.

### D. DISCUSSION

Our contour detector is generated by simulating the visual cortex of human beings. It is functionally similar to the visual cortex in contour extraction. The visual cortex is the product of millions of years of evolution. It is believed to be more advanced than any machine system in visual perception. There are many benefits that contribute to its superiority. Here, we only discuss those associated with the proposed contour detector.

First, by simulating the retina cell, our detector enhances the edges of an image by means of color opponency and spatial opponency. As a consequence, the strengths of edges in the image are amplified, which increases the differential of simple and complex cells in the visual cortex. Some edges with little strengths are protected by the opponency mechanism. Note that these low-strength edges may be parts of the contours. Hence, the opponency mechanism employed by our proposal helps improve the performance of contour detection by protecting the weak parts of the contours. No such mechanism is employed by DNN-based methods.

Second, NCRFs and sparse coding are employed by our contour detector to suppress textural details. There is physiological evidence that NCRFs and sparse coding are employed in the visual cortex [11]. From the viewpoint of information coding, only necessary data for a specific task are encoded for the sake of energy minimization. Those edges not reflecting object boundaries are textural details. They are similar to noises in a signal,and should be excluded from encoding. It is clear from FIGURE 3 that the contours produced by our detector are better than the contours produced by CO and HED in suppressing textural details. Our detector produces ideal contours, with cleaner and meaningful segments.

Third, contour is believed to be the intermediary between low-level edges and the high-level representation of objects (shape) in the visual cortex. It is the material for shape coding in the visual cortex. By employing the endstopped cells and curvature cells in area 4 of the visual cortex (V4), the main parts of contour are extracted from cluttered edges. Hence, our contour detector enhances the core of the contour while suppressing the textural details. However, neither the bioinspired CO and SCO nor the DNNs work in this way. We believe that the employment of endstopped cells and curvature cells is the right way to obtain real contours. This is

not the case for CO and SCO, since they only model simple and complex cells, which are only associated with low-level edge encoding.

In conclusion, contours are the parts of edges in an image that are projected from object boundaries. Contour detection is the act of extracting these parts from cluttered edges. Thus, a good contour detector protects the projections of object boundaries, and suppresses textural details as best as it can. From this perspective, our proposal is a good contour detector.

## VI. CONCLUSION

In this paper, we proposed a bioinspired contour detector by simulating the curvature cell in the visual cortex. The contour detector extracts the salience from an edge image, which is the core of the contour. It enables our detector to outperform the other bioinspired contour detectors, such as SO and SCO. Experiments on contour extraction and visual object classification showed that the proposal is both biologically sound and technologically useful. This implies that contour play an important part in the human visual perception system and is very useful in computer vision. However, the mechanism of the visual cortex is much more complex than the model we use here. We believe that more detailed and comprehensive investigations should be carried out in the future to improve the performance of contour detection.

## REFERENCES

[1] J. Shotton, A. Blake, and R. Cipolla, "Contour-based learning for object detection," in *Proc. 10th IEEE Int. Conf. Comput. Vis. (ICCV)*, vol. 1. Beijing, China, Jun. 2005, pp. 503–510.

[2] J. Shotton, A. Blake, and R. Cipolla, "Multiscale categorical object recognition using contour fragments," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 30, no. 7, pp. 1270–1281, Jul. 2008.

[3] A. Yilmaz, X. Li, and M. Shah, "Contour-based object tracking with occlusion handling in video acquired using mobile cameras," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 26, no. 11, pp. 1531–1536, Nov. 2004.

[4] K. Yang, S. Gao, C. Li, and Y. Li, "Efficient color boundary detection with color-opponent mechanisms," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2013, pp. 2810–2817.

[5] M. W. Spratling, "Image segmentation using a sparse coding model of cortical area v1," *IEEE Trans. Image Process.*, vol. 22, no. 4, pp. 1631–1643, Apr. 2013.

[6] K.-F. Yang, S.-B. Gao, C.-F. Guo, C.-Y. Li, and Y.-J. Li, "Boundary detection using double-opponency and spatial sparseness constraint," *IEEE Trans. Image Process.*, vol. 24, no. 8, pp. 2565–2578, Aug. 2015.

[7] A. J. Rodriguez-Sanchez and J. K. Tsotsos, "The importance of intermediate representations for the modeling of 2D shape detection: End-stopping and curvature tuned computations," in *Proc. CVPR*, Jun. 2011, pp. 4321–4326.

[8] D. Marr and E. Hildreth, "Theory of edge detection," *Proc. Roy. Soc. London. B, Biol. Sci.*, vol. 207, pp. 187–217, Feb. 1980.

[9] D. H. Hubel and T. N. Wiesel, "Receptive fields, binocular interaction and functional architecture in the cat's visual cortex," *J. Physiol.*, vol. 160, no. 1, pp. 106–154, Jan. 1962.

[10] H. B. Barlow, "Pattern recognition and the responses of sensory neurons," *Ann. New York Acad. Sci.*, vol. 156, no. 2, pp. 872–881, Apr. 1969.

[11] L. R. Squire, D. Berg, F. E. Bloom, S. Du Lac, A. Ghosh, and N. C. Spitzer, *Fundamental Neuroscience*, 4th ed. Salt Lake City UT,USA: Academic, 2012.

[12] H. Wei, B. Lang, and Q. Zuo, "Contour detection model with multi-scale integration based on non-classical receptive field," *Neurocomputing*, vol. 103, pp. 247–262, Mar. 2013.

[13] K.-F. Yang, C.-Y. Li, and Y.-J. Li, "Multifeature-based surround inhibition improves contour detection in natural images," *IEEE Trans. Image Process.*, vol. 23, no. 12, pp. 5020–5032, Dec. 2014.

[14] A. Akbarinia and C. A. Parraga, "Feedback and surround modulated boundary detection," *Int. J. Comput. Vis.*, vol. 126, no. 12, pp. 1367–1380, Dec. 2018.

[15] F.-P. Wang and P.-L. Shui, "Noise-robust color edge detector using gradient matrix and anisotropic Gaussian directional derivative matrix," *Pattern Recognit.*, vol. 52, pp. 346–357, Apr. 2016.

[16] W. Zhang, Y. Zhao, T. P. Breckon, and L. Chen, "Noise robust image edge detection based upon the automatic anisotropic Gaussian kernels," *Pattern Recognit.*, vol. 63, pp. 193–205, Mar. 2017.

[17] G. Wang, C. Lopez-Molina, and B. De Baets, "Multiscale edge detection using first-order derivative of anisotropic Gaussian kernels," *J. Math. Imag. Vis.*, vol. 61, no. 8, pp. 1096–1111, Oct. 2019.

[18] J. Canny, "A computational approach to edge detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. PAMI-8, no. 6, pp. 679–698, Nov. 1986.

[19] P. Dollar and C. L. Zitnick, "Fast edge detection using structured forests," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 8, pp. 1558–1570, Aug. 2015.

[20] J. Kivinen, C. Williams, and N. Heess, "Visual boundary prediction: A deep neural prediction network and quality dissection," in *Proc. 17th Int. Conf. Artif. Intell. Statist.*, vol. 33, 2014, pp. 512–521.

[21] S. Xie and Z. Tu, "Holistically-nested edge detection," *Int. J. Comput. Vis.*, vol. 125, nos. 1–3, pp. 3–18, Dec. 2017.

[22] Y. Liu, M.-M. Cheng, X. Hu, J.-W. Bian, L. Zhang, X. Bai, and J. Tang, "Richer convolutional features for edge detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 41, no. 8, pp. 1939–1946, Aug. 2019.

[23] J. He, S. Zhang, M. Yang, Y. Shan, and T. Huang, "Bi-directional cascade network for perceptual edge detection," 2019, *arXiv:1902.10903*. [Online]. Available: http://arxiv.org/abs/1902.10903

[24] A. R. A. Besari, R. Zamri, K. A. A. Rahman, M. D. M. Palil, and A. S. Prabuwono, "Surface defect characterization in polishing process using contour dispersion," in *Proc. Int. Conf. Soft Comput. Pattern Recognit.*, Malacca, Malaysia, 2009, pp. 707–710.

[25] C. Jian, J. Gao, and Y. Ao, "Automatic surface defect detection for mobile phone screen glass based on machine vision," *Appl. Soft Comput.*, vol. 52, pp. 348–358, Mar. 2017.

[26] L. M. Hurvich, "An opponent-process theory of color vision," *Psychol. Rev.*, vol. 64, no. 6, pp. 384–404, 1957.

[27] D. H. Hubel and T. N. Wiesel, "Receptive fields and functional architecture in two nonstriate visual areas (18 and 19) of the cat," *J. Neurophysiol.*, vol. 28, no. 2, pp. 229–289, Mar. 1965.

[28] S. Marcelja, "Mathematical description of the responses of simple cortical cells," *J. Opt. Soc. Amer.*, vol. 70, no. 11, pp. 1297–1300, Nov. 1980.

[29] J. Allman, F. Miezin, and E. McGuinness, "Stimulus specific responses from beyond the classical receptive field: Neurophysiological mechanisms for local-global comparisons in visual neurons," *Annu. Rev. Neurosci.*, vol. 8, no. 1, pp. 407–430, Mar. 1985.

[30] L. Maffei and A. Fiorentini, "The unresponsive regions of visual cortical receptive fields," *Vis. Res.*, vol. 16, no. 10, p. 1131, Jan. 1976.

[31] D. J. Field, J. R. Golden, and A. Hayes, "Contour integration and the association field," in *The New Visual Neurosciences*, J. S.Werner and L. Chalupa, Eds. Cambridge, MA, USA: MIT Press, 2014, pp. 627–638.

[32] S. Alpert, M. Galun, A. Brandt, and R. Basri, "Image segmentation by probabilistic bottom-up aggregation and cue integration," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 2, pp. 315–327, Feb. 2012.

[33] D. H. Hubel and T. N. Wiesel, "Receptive fields and functional architecture of monkey striate cortex," *J. Physiol.*, vol. 195, no. 1, pp. 215–243, Mar. 1968.

[34] F. Heitger, L. Rosenthaler, R. Von Der Heydt, E. Peterhans, and O. Kübler, "Simulation of neural contour mechanisms: From simple to end-stopped cells," *Vis. Res.*, vol. 32, no. 5, pp. 963–981, May 1992.

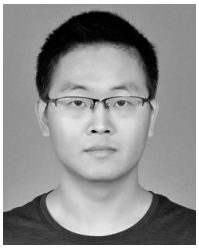
[35] H. Kato, P. O. Bishop, and G. A. Orban, "Hypercomplex and simple/complex cell classifications in cat striate cortex.," *J. Neurophysiol.*, vol. 41, no. 5, pp. 1071–1095, Sep. 1978.

[36] A. Dobbins, S. W. Zucker, and M. S. Cynader, "Endstopping and curvature," *Vis. Res.*, vol. 29, no. 10, pp. 1371–1387, Jan. 1989.

[37] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2005, pp. 886–893.

[38] C. J. C. Burges, "A tutorial on support vector machines for pattern recognition," *Data Mining Knowl. Discovery*, vol. 2, no. 2, pp. 121–167, 1998.

[39] D. Martin, C. Fowlkes, D. Tal, and J. Malik, "A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics," in *Proc. 8th IEEE Int. Conf. Comput. Vis. (ICCV)*, Jul. 2001, pp. 416–423.
[40] P. Arbeláez, M. Maire, C. Fowlkes, and J. Malik, "Contour detection and hierarchical image segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 5, pp. 898–916, May 2011.
[41] H. Xiao, K. Rasul, and R. Vollgraf, "Fashion-MNIST: A novel image dataset for benchmarking machine learning algorithms," 2017, *arXiv:1708.07747*. [Online]. Available: http://arxiv.org/abs/1708.07747
[42] O. J. O. Soderkvist, "Computer vision classifcation of leaves from Swedish trees," M.S. thesis, Linköping Univ., Linköping, Sweden, 2001.
[43] *Fashion-MNIST Benchmark*. Accessed: Feb. 5, 2020. [Online]. Available: http:// fashion-mnist.s3-website.eu-central-1.amazonaws.com
[44] *Fashion-MNIST Benchmark CNN*. Accessed: Feb. 5, 2020. [Online]. Available: https://github.com/zalandoresearch/fashion-mnist
[45] H. F. Eid, E. Al-Azhar UniversityFaculty of ScienceCairo, and A. Darwish, "Variant-order statistics based model for RealTime plant species recognition," *Int. J. Inf. Technol. Comput. Sci.*, vol. 9, no. 9, pp. 77–84, Sep. 2017.
[46] H. F. Eid and A. Abraham, "Plant species identification using leaf biometrics and swarm optimization: A hybrid PSO, GWO, SVM model," *Int. J. Hybrid Intell. Syst.*, vol. 14, no. 3, pp. 155–165, Mar. 2018.
[47] D. G. Tsolakidis, D. I. Kosmopoulos, and G. Papadourakis, "Plant leaf recognition using Zernike moments and histogram of oriented gradients," in *Hellenic Conference on Artificial Intelligence*. Ioannina, Greece: Springer, May 2014, pp. 406–417.
[48] D. Kuang, "A 1d convolutional network for leaf and time series classification," 2019, *arXiv:1907.00069*. [Online]. Available: http://arxiv.org/abs/1907.00069



**ZEKUN CHEN** received the B.Eng. degree from Fujian Jiangxia University, Fuzhou, China, in 2015. He is currently pursuing the M.Eng. degree in information and communication engineering with Fujian Normal University, Fuzhou. His research interests include image processing and machine learning.

**RONGTAI CAI** received the B.Eng. degree from Jilin University, Changchun, China, in 2003, and the Ph.D. degree from the Graduate University of Chinese Academy of Sciences, Beijing, China, in 2008. He was a Visiting Assistant Professor with the University of Oxford, Oxfordshire, U.K., from 2014 to 2015. He is currently an Associate Professor with the College of Photonic and Electronic Engineering, Fujian Normal University, Fuzhou, China. His research interests include computer vision and machine learning, cortex-like visual computation, and the neural computation mechanisms underlying object recognition.