Hypercolumn-array Based Image Representation and Its Application to Shape-based Object Detection

Hui Wei[[1]](#footnote-1), Zheng Dong[[2]](#footnote-2), and Bing Liu[[3]](#footnote-3)

April 11, 2016

**Abstract**

Machine learning is perhaps the most popular strategy currently employed in object recognition. The strategy typically uses a high-dimensional vector to describe an image, and applies some classification algorithm to isolate positive areas apart from negative areas. While the advantage of this strategy is its architectural simplicity, it simultaneously suffers from computational complexity, fragile representation, unguaranteed generalization, black arts in optimization, and so on. In comparison with low-order features, such as color or gradient, contour is more stable and persistent; therefore, shape-based methods take account of a greater number of essential aspects for object recognition. Biological and psychological evidence increasingly reveals that geometrical and topological features are the keys to object recognition. Although these types of high-level features are not easily obtained, their discriminating abilities greatly improve the efficiency of perception. Attracted by the excellent performance of neural visual systems for shape processing, we simulate the mechanism of hypercolumns in the V1 cortex of mammals that selectively responds to bar stimuli, and design an orderly-arranged array to extract and represent all possible linear or near-linear stimuli in an image. Each unit of this array can cover all orientation stimuli occurring over a certain small area. These effective units together produce a type of low-dimensional vector to describe shape. Based on the neighborhood of units in the array, we construct a graph whose node represents a short line segment with a certain position and slope. Therefore, any segment of contour, i.e., a curve, in an edge image might be a route in that graph. The most significant thing here is that a graph converts an image, comprised of typically unstructured raw data, into structured and semantic-enriched data. We search along the routes in that graph and compare them with a shape template for object detection. Organizing segments of contour into a graph greatly upgrades the level of image representation, remarkably reduces the load of combinations, significantly improves the efficiency of object searching, and facilitates the intervening of high-level knowledge. This work provides a systematic infrastructure for shape-based models.

Keywords: orientation column, shape representation, object recognition, primary visual cortex.

# Introduction

Detecting an object within a complex background is still one of the major challenges of pattern recognition and computer vision. The difficulty lies in the conditions of image capture, where images are captured under different illumination conditions, from different distances, and, sometimes, with overlapping objects. These variations make an object’s appearance, either in size, in posture, or in color, endless changing. While machine learning is arguably the most popular strategy currently employed in object recognition, the uncertainties associated with an ever changing environment and limited training create numerous difficulties for machine learning methods to achieve efficient object recognition. As such, obtaining an efficient, stable, and convenient method of representing features becomes the key requirement of object recognition in the real-world. The most stable characteristic of an object is its contour. The shapes or contours of most rigid objects tend to remain invariant regardless of a changing environment. Therefore, an object’s shape or contour represents the most promising feature for efficient object recognition.

The primary visual cortex, which is also known as V1, locates in the striate cortex of mammals. It is the best-studied visual area in the brain. In this paper, we simulate the neural mechanism of V1 because this area contains many orderly-arranged hypercolumns that represent the orientation information of a visual stimulus. This biological infrastructure provides a feasible entity to record object contours. We therefore design and train a self-organizing map (SOM) to extract the line information of an image. Under this approach, any line, regardless of its position, length, or slope, will activate one or more orientation sensitive neurons, so an edge-image will be re-described by a set of active neurons. Based on this line-representing platform, we propose a contour-matching algorithm that uses geometrical characteristics to recognize objects. This provides good performance in environment adaptation and better generalization.

The remainder of this paper has been organized as follows. Section 2 reviews some related studies in shape-based object recognition and visual neurobiological mechanism simulation. The advantages and disadvantages of the main features of each study are briefly discussed. Section 3 presents the training process involving orientation columns, and how an image is represented by an array of artificial orientation columns. Section 4 describes how an edge image is converted into an undirected graph, and how route searching is conducted for this graph. Section 5 presents the method of similarity matching between routes and templates, and how this method is applied to the recognition of objects in real images. Experimental results are presented in Section 6, which is divided into two parts. Part one analyzes the capability for image representation and reduction provided by the proposed array of artificial orientation columns. Part two presents the recognition results obtained for some real image datasets. The final section presents conclusions and future expectations.

# Related Work

Here, we divide the discussion of related work into two components. One component concerns the modeling of the V1 area, whereas the other component is concerned with shape- or contour-based object recognition methods.

## V1 simulation research

A number of approaches to the computational modeling of the neural mechanism of the V1 area of mammals have been proposed. The differences between these approaches lie in their points of focus, where some approaches focus on pathway, some on mechanism, and some on computation. Some studies have concentrated on the visual neural pathway [1, 2, 3, 4]. For example, Bickle et al. [1] proposed an artificial neural network with interactive activation and competition (IAC) mechanism by reference to the lateral geniculate nucleus (LGN), which is the direct pathway from the primary visual cortex to the thalamus, to implement several functions of the visual cortex. Shariati et al. [2] simulated the entire neural pathway from a photoreceptor in the retina to the second sub-layer of the primary visual cortex. Some studies have concentrated on visual neurocomputing problems [5, 6, 7, 8, 9]. For example, Curuklu [5] proposed a Bayesian confidence propagation neural network (BCPNN) to simulate a hypercolumn, which is the functional unit of the visual cortex. Wang et al. [6] used sparse coding and a back-propagation (BP) neural network to model the primary visual cortex. Finally, a number of studies have concentrated on the functional roles of variant neurons in the visual system [10, 11, 12, 13, 14, 15, 16, 17, 18]. Okamoto [10] designed a honeycomb-like model based on a mechanism of short-range horizontal connections to explain the topological characteristics of the V1 area. Willmore [11] successfully simulated the visual cortex based on Hebbian and anti-Hebbian rules. Bednar et al. [12, 13] implemented a gain-controlled, adaptive, and lateral-connected model that specialized the function of every neuron in the pathway, and simulated the primary visual cortex more completely. The above studies were able to realize some neural circuits and partial functions of the V1 area. However, they did not sufficiently account for real image representation, which is the ultimate and most important function of the V1 area. In fact, all subsequent processing, such as object recognition and scene understanding, begins in the V1 area. Above all, these studies lacked the crucial consideration of how the V1’s output meets the demands of subsequent recognition tasks. This constitutes one of the main goals of the present study.

One of the fundamental functions of V1 area is extracting features about edges and orientations from the visual stimuli. The function is carried out by the orientation columns in V1, which is simulated with the hypercolumn array in the proposed model. Many computational methods in image processing also achieve similar edge detection function [19, 20, 21]. Canny detector [19] and the globalPb [20] generate edge maps, which represent edges in the level of pixels. The tensor voting [21] and the LSD method [22] detect line segments in images. These representations provide discrete information about pixels and edges, which is not sufficient for shape detection or object recognition. Moreover, these methods usually involve complicated computation, which is difficult to be implemented with neural circuits for real-time computer vision tasks. The proposed hypercolumn-array follows the structure of the neural system, which is efficient and works seamlessly with the succeeding component for shape-based object recognition.

## Shape- or contour-based object recognition

Shape- or contour-based object recognition research has a long history. In the early stage, chamfer matching [23, 24, 25, 26] was a frequently-used method that transformed an original image into a distance map, and searched for the point with the best match. This method demonstrated a certain adaptability. As an additional aid, an image pyramid was designed to achieve scale invariance [23]. However, this method usually exhibits high time complexity and has a low tolerance to background interference. To address this problem, several new variants of the chamfer matching method have been proposed in recent years. For example, Shotton [24] designed an oriented chamfer matching model, Thayananthan [25] combined this method with a shape context descriptor, and Opelt [25] added AdaBoost. These variants were observed to improve the efficiency and recognition rate.

Another type of shape-based method is mainly based on contour segment matching between templates and real images. This type of method firstly converts an image into edges, and then fits curves or line segments to the edge pixels. Subsequently, the set of curves or line segments are represented with some geometrical feature descriptors. Finally, the similarity between the set of segments and shape templates are obtained. Once matching pairs between line segments and the template are established, some guesses can be made to estimate the possible positions for the target object, and then verify these possible positions one-by-one. This type of method can be further divided into two categories. One is the dominant-set method and the other is the partial-match method. Pavan [27] proposed the dominant-set method for finding the principle corresponding components in two sets. Yang [28] used this method to form a scale invariant global shape similarity estimation to match between template contour parts and image edge fragments. Partial-match methods [29, 30, 31] are similar to dominant-set methods. Ma [29] obtained a similarity matrix based on the angles of edges in a contour and maximized the similarity between classes through Liu’s cluster algorithm [31]. Objects were then recognized by seeking the largest community. Differing from the work of Ma [29], in the stage of evaluation, Riemenschneider [30] estimated the object position using the pyramid matching kernel (PMK) algorithm after conducting contour segment matching. Methods proposed in [32, 33] introduced a shape context descriptor. Zhu [32] used linear programming to find a one-to-one mapping between the image and the template. Srinivasan [33] developed this method further by introducing a bottom-up process to optimize many-to-one mapping results. In general all these methods exhibit high time complexity subject to exhaustive comparison between real-world images and templates.

Other shape-based recognition methods have been proposed, such as the constellation model [34] and spectral model [35]. These methods use the relative location of points in a contour, where the established spatial relationships are regarded as constraints. The spatial relationships of an edge point to its surroundings are calculated in order to determine whether the surrounding pixels belong to the target object or to the background. A sparse representation is thus derived from these spatial relationships, and a probabilistic inference is used to estimate the position of a target object. The main disadvantage of this type of method is that the recognition result can be greatly affected when the foreground and the background are mixed or objects overlap. Leordeanu [35] proposed a spectral model to calculate the geometrical relationships of pairwise points, and to analyze where pairwise points were densely located in a contour. The drawback of this model is that it requires a great many training samples.

In addition, some shape-based recognition methods are a mixture of several existing approaches. Fergus [34] counted the occurrences of foreground features, and consulted with contour matching to obtain the probabilities of foreground to background, and, finally, to estimate the position of a target object. Gupta [36] designed a contour descriptor by reference to the concept of torsion in physics. This descriptor performs well for changing scales. However, the method performs poorly when the contour is somewhat missing because it is concerned with global contours. Schlecht [37] described the marked features of an object by a codebook of angles, and combined Hough voting for object detection. Ferrari [38] proposed gradual contour searching to recognize objects in a real-world image.

In general, shape-based object recognition methods are not covered by the mainstream technology. The reasons for this are as follows. (a) This type of method always requires good preprocessing to reduce noise or to obtain some basic geometrical units such as line segments; however the preprocessing results are affected by numerous factors. (b) A target object’s background, position, size, and posture are changing constantly, which is a substantial challenge for template matching. (c) Template matching is essentially a combinational optimization problem that requires searching for a globally optimum solution, which is a severe challenge for any searching algorithm. (d) The template matching method involves many process steps such as template acquisition, template formal representation, search-friendly image re-organization and re-representation after preprocessing, definition of geometrical features, searching and combining geometrical features, definition of geometrical constraints, and generating and verifying assumptions. All these requirements increase the complexity of a complete shape-based system.

# Hypercolumn-array Based Image Representation

## The structure of a hypercolumn-inspired array

In [39, 40, 41], a multi-layer neural computational model that simulated hypercolumns in the primary visual cortex was proposed. The output layer of this model can be used to represent the orientation information of a contour image.

As shown in Figure 1, this layer is comprised of a number of large, hexagonal hypercolumns, and two of these in the upper-right corner of the figure are marked by two black frames. Each hypercolumn consists of nineteen orientation chips (small hexagons). Each chip can selectively respond to an orientation specific segment of contour existing in its receptive field. Figure 2 shows the scope and the distribution of the receptive fields of several neighboring hypercolumns, and how they share these receptive fields. All nineteen chips in a single column can completely cover the range of slope from 0 to *π*. All these columns assemble into an array, which plays the roles of extraction and representation of the orientation features of contour in an image.

## Representing images by a hypercolumn-inspired array

One of the most important discoveries in neurobiology is that a hypercolumn in the primary cortex can respond to a series of stimuli with continuously changing orientations. As the discoverers, Hubel and Wiesel won the Nobel Prize in 1982. We were therefore inspired to use this mechanism to obtain and represent segments of contour in an image. When an edge image is presented to the aforementioned hypercolumn array, in a certain receptive field, some linearly distributed pixels will activate a certain orientation chip of a hypercolumn. This indicates that we can represent all contour information in an image by a set of active units (i.e., orientation chips). In mathematics, each unit represents a line vector. Therefore, the output of a hypercolumn array is no longer a group of discrete, isolated, and pixel-wise points, but some preliminarily-integrated short line segments that contain a wealth of geometrical information such as position, length, and angle. Thus, the hypercolumn array not only upgrades the representation level of an edge image, but also reserves the original characteristic of pixel distribution in an edge image. Figure 3 shows several examples of such representation. A simple triangle, an onion shape, and a contour of a car were presented to the hypercolumn array, which generated a set of activated orientation chips. This is very similar to using sparse coding to represent an image.

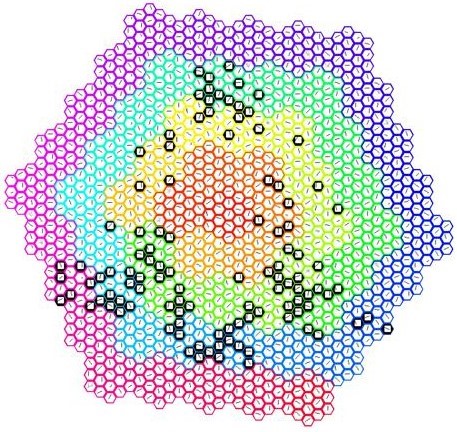


Figure 1: The structure of a hypercolumn-array, which is used for contour representation. Each small hexagon represents an orientation chip that corresponds to a simple cell. Each simple cell in the V1 area is sensitive to a bar of light with a particular orientation (slope). We evenly divide the overall orientation interval from 0 to *π* into nineteen sub-intervals. Nineteen orientation chips, each of which responds to one, and only one, sub-interval, are combined as a hypercolumn, and nineteen small hexagons happen to form a large hexagon. All the chips in the same hypercolumn share the same receptive field.

The foregoing examples indicate that an edge image can be easily represented with activated orientation chips in a hypercolumn array. Each activated chip represents a vector reflecting the position, length, and angle of a short line segment. In addition, the chip’s activation level can also indicate the level of similarity between a real stimulus and the chip’s built-in preference. This is equivalent to a comparison between an input and a template. For example, the stimulus in a receptive field might sometimes be a curve or a dotted line rather than a perfect straight line. Therefore, the output of hypercolumns is an approximation to the original edge image, as shown in Figure 4. This approximation can reflect the distribution of pixels in a receptive field, which offers a certain degree of flexibility and error tolerance.

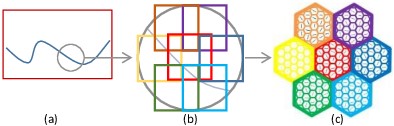


Figure 2: Seven neighboring hypercolumns and their receptive fields in a visual field. (a) A segment of contour in a visual field. (b) The purple rounded region is covered by seven square receptive fields, and they overlap with each other to some extent. (c) Seven colored neighboring hypercolumns whose corresponding receptive fields with the same colors are shown in (b).

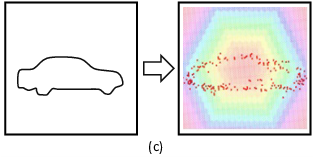
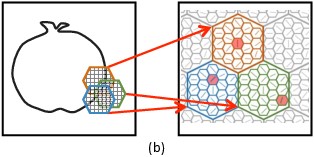
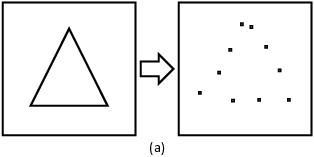


Figure 3: Examples of hypercolumn-array representation. (a) Some orientation chips (black points at the right) are activated by a simple triangle. The topological feature of the triangle is preserved by the set of active orientation chips. (b) The partial contour at the bottom-right of an onion from the MPEG7 dataset passes through three neighboring and overlapping receptive fields, which activates three orientation chips in the three hypercolumns respectively, and these chips correctly reflect the geometrical information in this contour segment. (c) A contour of a car is represented by an array with 452 hypercolumns. The extent to which the details of the car that are preserved increases with an increasing number of activated orientation chips.

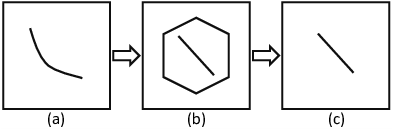
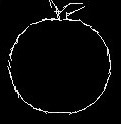
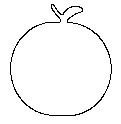


Figure 4: Orientation chips can approximately represent stimulus. (a) A real stimulus in a receptive field might be a small curve. (b) An orientation chip with the closest built-in orientation preference is activated. (c) The final output is an approximation of the original stimulus.

## Image reconstruction from an activated array

However, the scale of a hypercolumn-array, given as the number of hypercolumns in it, and the number of orientation chips in a hypercolumn, is limited. Thus, using a finite resource to represent infinite stimuli must inevitably result in approximation, and the degree of approximation must be evaluated after converting an original image to some activated orientation chips. To verify that the hypercolumn array can represent the original stimulus with sufficient fidelity, we rebuild an image by redrawing lines in the receptive fields of the activated orientation chips, and then compare the original image to the rebuilt one. As shown in Figure 5, the contour information of an original image can be effectively recorded, and the rebuilt version is nearly identical to the original. Further experimental verification of the image reconstruction ability of a hypercolumn-array will be presented in Section 6.



(a) Original image (b) Rebuilt image

Figure 5: The result of image reconstruction. (a) An original image from the MPEG7 dataset. (b) A rebuilt image based on the orientation chips activated by (a). Here, (a) and (b) are nearly identical.

# Processing From a Hypercolumn Array to a Graph

After the preprocessing of the hypercolumn array discussed in the previous section, an ordinary edge image has been converted to a group of active units, each of which represents a short line segment with a certain position, length, and slope, given as a quad vector: *x*-coordinate of the center point, *y*-coordinate of the center point, length, and slope. This representation by a group of units is more compact and information-rich. Compared with a complete contour, the line segment represented by an active unit is somewhat fine, and the information obtained from active units must be aggregated in some way to facilitate the subsequent recognition operation. The use of sets is a simple aggregation approach, but this approach is not sufficiently powerful to describe the topology of active units distributed in an array. The use of graphs is a more effective approach to describe the relations between adjacent or connected active units. Moreover, the layout characteristics of chips arranged in a hypercolumn as well as hypercolumns arranged in an array coincide with the functionality of graphs, where nodes and edges are two basic elements for constructing a graph. The analogy between a hypercolumn-array and a graph is quite a natural association. This suggests the application of graphs to describe the topological distribution of activated orientation chips.

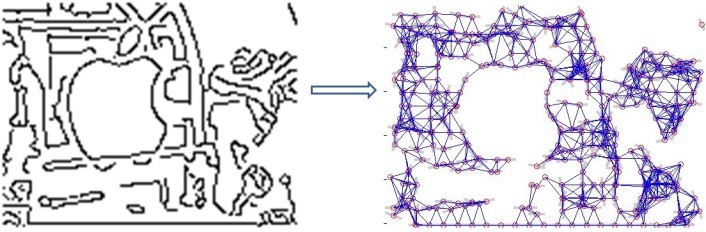


Figure 6: An edge image from the ETHZ dataset (at left) has activated some orientation chips. We can generate an undirected graph (at right), of which the nodes are these activated chips.

## The generation of a graph

We wish to represent a continuous contour by coordinating the information derived from multiple chips through the implementation of a graph. Therefore, the best manner of defining a graph for a hypercolumn-array is to set the chips as the nodes of the graph, and establish the edges between those nodes (i.e., chips) that might be successively joined together to represent a slightly longer line or curve. Because all 19 chips in a given hypercolumn share a completely coincident receptive field, and multiple lines are unlikely to exist concurrently in this fairly small area, these chips are usually not activated simultaneously. Therefore, no edges can be established between these independent nodes. Only those chips with neighboring or partially overlapping receptive fields might be coordinated to represent a contour that passes through multiple receptive fields. Links are assigned between these possibly coordinated nodes that usually belong to separate but adjacent hypercolumns. The present description defines a basic graph of chip nodes, i.e., an orientation chip graph. Some nodes of this graph will be activated when an image is presented to the hypercolumn array. Most importantly, the orientation chip graph provides a structural data form for subsequent geometrical or topological analysis, as illustrated in Figure 6.

## Graph searching and route-parsing

The most significant value provided by the graph of chip nodes is (a) it effectively organizes raw image data, which is converted into an ordered structure, and (b) it facilitates the process of searching object contours. As indicated by the previous discussion, representing a continuous contour requires the coordination of multiple chips, which correspond to multiple connected nodes in our graph. From the point of view of a graph, the sequence of these connected nodes forms a route. Figure 7 shows an example of a route. The key to shape-based object recognition involves locating one or several such routes that represent the contour or partial contour of the object to be discriminated. Therefore, a route-searching algorithm is required.

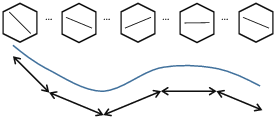


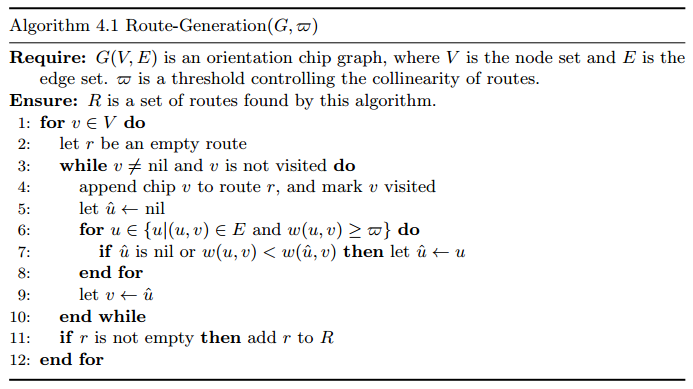
Figure 7: A segment of an object’s contour might be a route of a chip graph. The top is a sequence of chips linked together by a route; the middle is the original stimulus that activated the above route; and the bottom is the approximate curve rebuilt from that route.

Route-searching seeks to combine the neighboring nodes of a graph into routes that represent contour segments. However, some difficulties are encountered in this process because the contour of an object might suffer from interference with its background and be mixed with noise or be broken into several shorter segments. As is shown in Figure 5, the connectivity of a graph constructed from activated orientation chips can be high. A randomly chosen long route might go through both an object and its background. Ideally, a route should be either part of the object’s contour or from the background, and also it should convey enough information for us to determine whether it is part of the object’s contour. Therefore the searching process must be carefully guided to yield a reasonable result. Natural contours tend to be smooth and continuous, which provides a good hint. We assign weights to edges to measure the smoothness and continuity between chip nodes. Given two adjacent nodes  and  in the chip graph, the weight  of the edge  is defined as follows.

(1)

In this equation,  is the short line segment represented by the corresponding orientation chip and  is the orientation of that chip. The function  calculates the minimal distance between the endpoints of two line segments. It measures the continuity between the two chips. The difference of orientation is multiplied by a factor  to measure the smoothness. In this paper, the orientation is measured in degree and  takes . The edge weights range from  to . Larger weights indicate stronger collinearity between chip nodes. Figure 7 illustrates weight setting in different cases.

In the following Algorithm 4.1, the route searching process is guided by edge weights. A threshold  for edge weight is used to control the collinearity of the routes. In this paper,  takes . This threshold ensures that the distance between line segments represented by adjacent chips in a route are within 3 pixels and the orientation difference is less than .



Algorithm 4.1 roughly provides some basic routes out of a graph that can be rebuilt to obtain some continuous and smooth segments of contour. However, there are some additional opportunities for those short basic routes to join together because the searching process may start from the middle of a longer route, which is broken into two segments. Here, we use Algorithm 4.2 to link these broken routes.



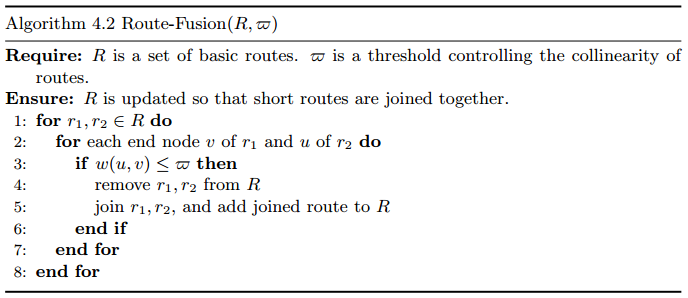
(a) *w* = *e*−max{0*,*0} = 1 (b) *w* = *e*−max{0*,*15*λ*} = 0*.*11



(c) *w* = *e*−max{3*,*0} = 0*.*05 (d) *w* = *e*−max{0*,*35*λ*} = 0*.*01

Figure 8: Setting the edge weight between two orientation chips. A larger value indicates a tighter or more collinear connection. (a) If two chips represent a longer line cooperatively, then the weight between them equals 1. (b) If two chips represent a smooth curve, then the weight is relatively large. (c) If two chips represent discontinuous short line segments, then the weight is small. K: If two chips represent a sharp turn, then the weight is small.

Because the number of basic routes is much less than the number of nodes, Algorithm 4.2 usually requires less time than Algorithm 4.1. After the fusion process, some longer routes are obtained. Figure 9 presents an example. From this example, we can understand that the process of long-route formation can provide further integration of the edge information of an image. These long routes are usually lines or smooth curves. These routes have geometrical significance because they are no longer discrete pixels, and may include all or a part of the contour of an object. Therefore, object recognition can be realized by matching long routes with a template. In this phase of the process, the number of long routes is greatly reduced. The succeeding object recognition process is thus an operation involving a small set of curves, which is inevitably more efficient than an operation involving a large set of pixels.



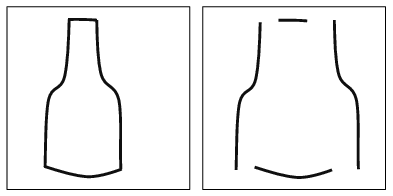


Figure 9: After conducting Algorithms 4.1 and 4.2, the contour of an object (at left) can be denoted by several routes (at right).

# Route-based Shape Matching

With the algorithms described in the previous section, a set of routes are extracted from the orientation chip graph. These routes consist of sequence of orientation chips representing continuous and smooth contour segments. We can evaluate the similarity between these routes and the shape template. If a sub-graph contains many routes of high similarity to the shape template, then it is highly possible that an object of that shape is located in the corresponding sub-region of the image which the graph represents. Object detection based on this idea requires two steps: the comparison of the similarity between a route and a segment of the template, and the assessment of the overall similarity between a sub-graph and the whole template.

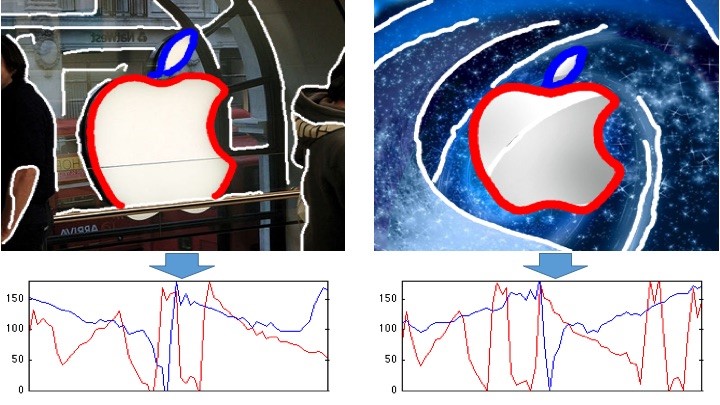


Figure 10: Representing routes with vectors. The routes in the images are transformed into vectors plotted below the images.

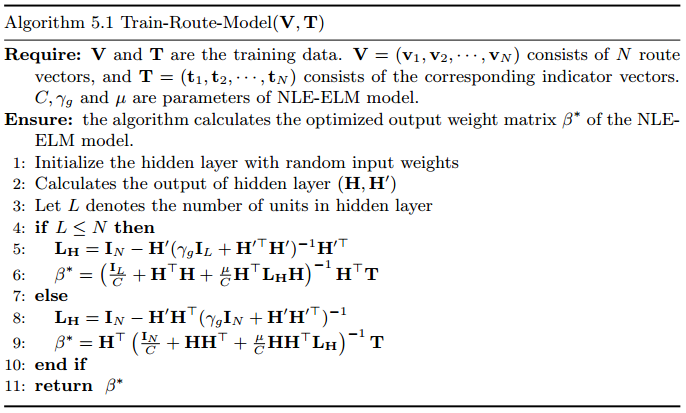
## Measurement of similarity between a route and a template segment

A number of algorithms exist at present for conducting curve similarity comparison. In [42], a curve representation method has been proposed based on a sequence of tangent angles, but this method is not appropriate for measuring similarity between a partial contour segment and a complete shape template. In [43], a type of turning function has been proposed to represent polygons and to measure their shapes, but this method is very sensitive to small zigzag or burr noise. It is not suitable for incomplete contours or contours mixed with the background. In [44], the dynamic time warping (DTW) method has been adopted to curve alignment, but his method is not applicable to curves that cannot be presented with one dimensional functions.

In this paper, the curve similarity is measured with a machine learning approach based on the non-linearly embedded extreme learning machine (NLEELM) [45]. The extreme learning machine (ELM) [46, 47, 48, 49] is a fast training algorithm for single hidden layer feedforward neural networks (SLFNNs). The NLE-ELM algorithm improves the performance on high-dimensional and sparse data. It is essentially a kind of self-embedding ELM, where inner-layer ELM is used for dimensionality reduction and outer-layer ELM is trained for classification or regression based on inner-layer ELM. It inherits the merits of ELM, such as avoiding local optima, fast learning speed.

The routes are represented with vectors which are provided as input to the NLE-ELM algorithm. By definition, a route is intrinsically a sequence of activated orientation chips. Therefore, it is straight-forward to represent a route with a vector. Each element of  is the orientation of an activated chip in the route. Examples of the vectors are shown in Figure 9. In each image, two routes around the contour of the apple logo are chosen to be represented into vectors of orientation.

We train an NLE-ELM model to recognize routes from a number  of object categories. For each route vector, the desired output is a binary indicator vector, where  indicates the route belongs to the -th object category. For each object category, positive routes are chosen from the training samples within the ground-truth bounding boxes, and negative ones are chosen from the background. The training data is generated from these routes. The training algorithm is listed in Algorithm 5.1.



Given the aforementioned route representation and NLE-ELM model, we can measure the similarity between a route from the test image and a route from a template. Let  and  denote two routes and  and  the vector representation of the routes. The output function of the NLE-ELM model is, where  is obtained with Algorithm 5.1. The distance between routes  and  is defined as the Euclidean distance between the NLE-ELM output.

(2)

Figure 11 presents an example of a similarity comparison between two curves, in which *A* is a template, *B* is a curve with small zigzag noise, and *C* is a smooth curve. A subjective evaluation of the figure indicates that *B* and *A* are more alike than *C* and *A*. Although *B* has noise, its overall shape is more similar to *A*. The output of the similarity comparison algorithm exactly reflects this observation. The basic principle underlying our algorithm is that the holistic tendency in a curve is more significant for similarity measurements.

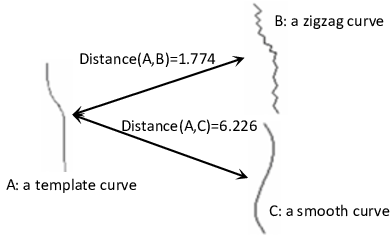


Figure 11: An example of a curve comparison based on a holistic strategy.

## Evidence accumulation

Based on the similarity measurement method described in the previous subsection, we can enumerate the long routes determined from the chip graph one by one. In most cases, they will contain some discriminative segment of an object contour, and we can compare them with a template and label them according to their similarity values. This is right an evidence collecting process, if we see object recognition as an inference process. If more than a single evidence assembling at some local area of an image is observed, then we have reason to believe that this area deserves further searching.

A graph generated as discussed in sub-section 4.1, describes neighborhood relations between short line segments, but the number of nodes in this graph is usually great. According to sub-section 4.2, a graph can be decomposed into some routes, each of which represents a curve. This middle-level integration can greatly improve the compactness of image representation, and, simultaneously, create a smaller searching space. For instance, as shown in Figure 12, based on 40 apple images in the ETHZ dataset, we respectively count the number of pixels in their edge images, the number of short line segments after hypercolumn array representation, and the number of routes after graph decomposition. The number of routes is exponentially less than the number of line segments and edge pixels. We can see that the degree of image compactness is steadily improved, and the searching space defined by routes has been greatly reduced.

If we assign a route as a node, and assign the neighborhood between the ends of two routes as a link, then we can generate a new type of graph with its grain-size enlarged. If the hypercolumn array based graph is denoted as a Line-graph, then this new graph is denoted as a Route-graph. For all nodes in the Route-graph, we define a distance matrix to measure the similarity between any pair of route curves and template curves. Figure 13 illustrates the formation process of this matrix. From each row, we can choose one or several routes with small values as evidence by which the hypothesis of a template curve occurring is supported. If the assemblage of evidence, i.e., those matched routes, distributes in a sufficiently narrow area, or if the routes are well-joined, then a goal object can be expected to occur in this area.

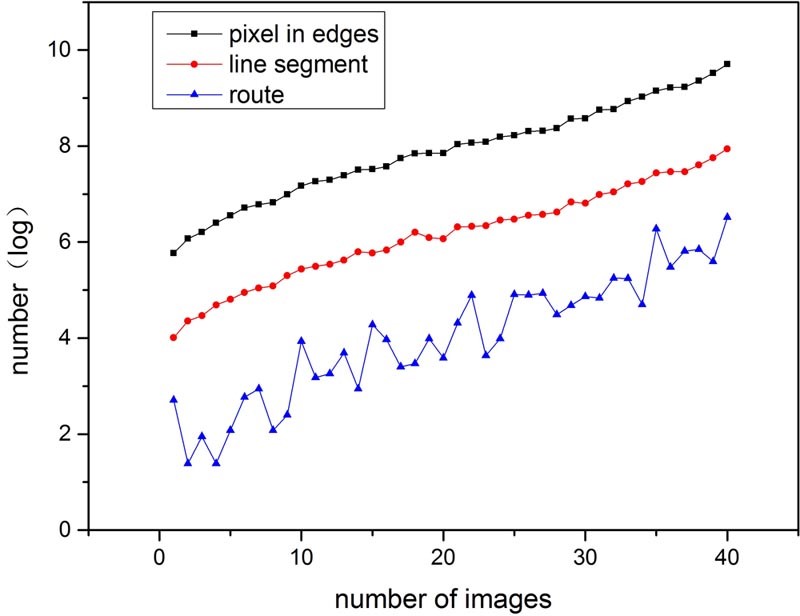
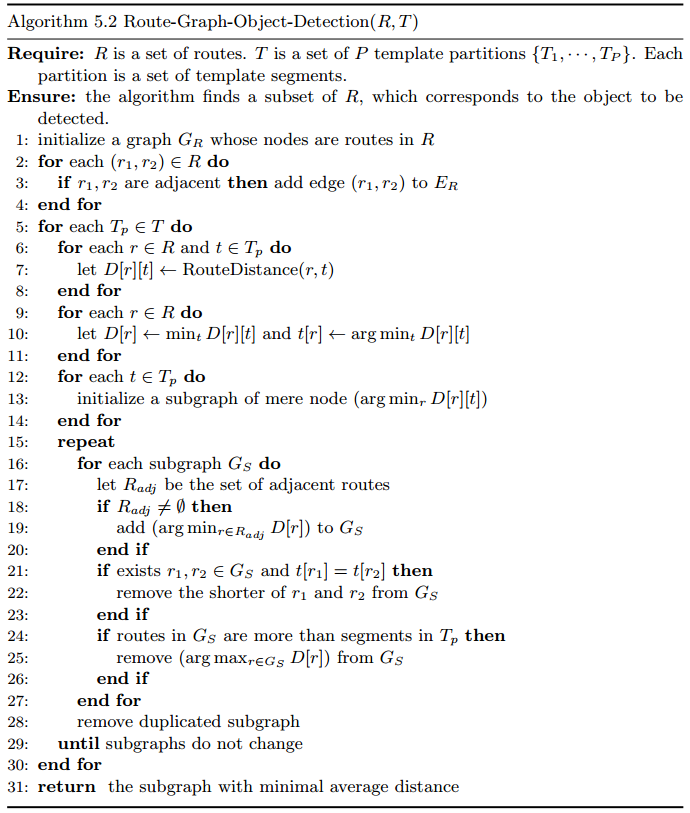


Figure 12: The Route-graph significantly reduces the searching space.

A single factor of evidence is not sufficient for sound discrimination because this matched curve might be a part of the background. The co-relationships of a group of curves are more important than any single curve’s similarity. For example, several curves that are very similar to some segments of an object’s contour may be scattered over an image, but their relative spatial relations might not satisfy the structural constrains of that object. In this paper, we adopt two sets of independent contour partition strategies for the same template. This guarantees that the contour segments can overlap with each other to some extent. The advantage of ensuring some measure of overlap is that the joining of two adjacent segments in one set can be established by a continuous segment in another set. We can use one set for template matching, and the other set for cross validation. Figure 14 illustrates this idea.

Based on the Route-graph, the following Algorithm 5.2 realizes a searching companied matching. Its basic flow is as follows. (a) Based on the similarity between a route and template, a sub-graph that contains dense evidence features is detected, which corresponds to a likely occurrence of a goal object. (b) This sub-graph’s route is joined with its neighboring route, and, if this new route exhibits an improved similarity with the template, then the sub-graph is updated by including the neighboring route. (c) A leaf node is deleted from a route, and, if this improves the similarity with the template, then the sub-graph is updated by deleting that node. (d) If any two neighboring routes in the subgraph have two corresponding curves in the template, validate whether these two segments are adjacent or not. If this topological constraint of connectivity is violated, then update the sub-graph by deleting the shorter route. (e) Carry out cross-validation by altering another set of contour partition strategy.



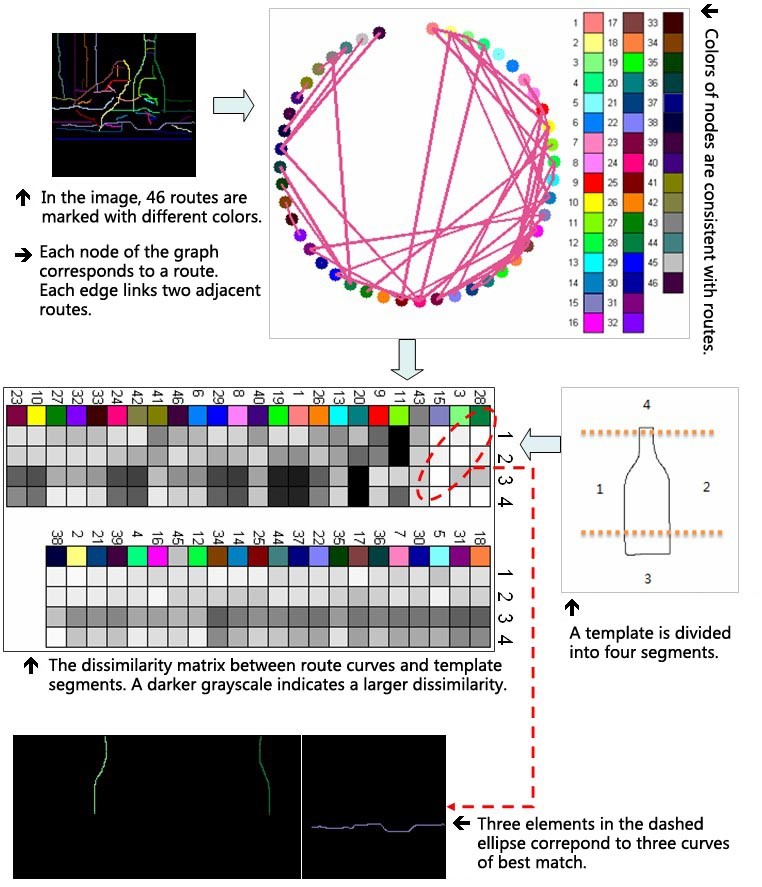


Figure 13: The distribution of evidence features among a Route-graph.

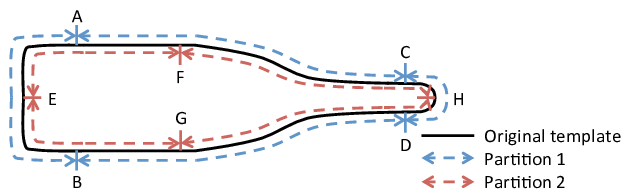


Figure 14: Two independent template partitions provide cross-verification.

# Experimental Results

## Image representation test

Our first experiment seeks to determine whether the hypercolumn-based array is able to preserve the major contours in an original image. For this test, we selected the well-known contour testing dataset, MPEG-7, consisting of 1400 images. Similar to the method proposed in [39], we also define the differences between the benchmark and the hypercolumn array’s output as noise, and the signal-to-noise ratio (SNR) as.

The statistical curves shown in Figure 15, where the red points are our SNR values and the blue points are the SNR values obtained from the gPb algorithm [51], indicate that our SNR results are better than those of the gPb algorithm. Figure 16 demonstrates three reconstruction examples, in which column (a) includes original images from the ETHZ dataset, column (b) includes corresponding edge images, and column (c) includes re-descriptions of the edge images using routes. Figure 17 is an example of the process of adding routes one-by-one, from which we observe that an image has been reduced to a small number of routes, and the route becomes the basic unit being manipulated.

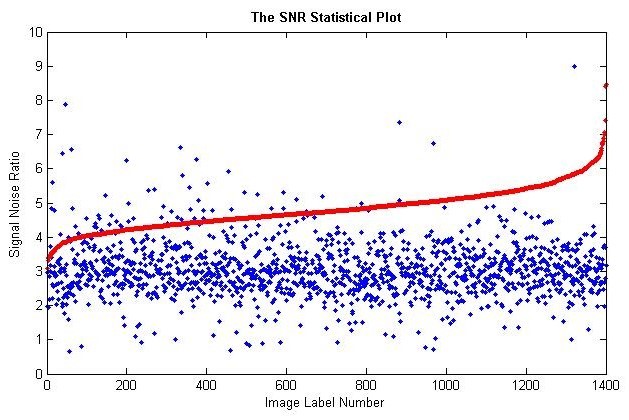


Figure 15: SNR result for the proposed method applied on the MPEG-7 dataset.

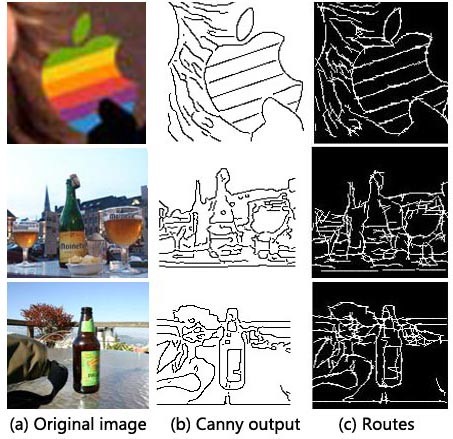


Figure 16: Representing images in the ETHZ dataset by routes.

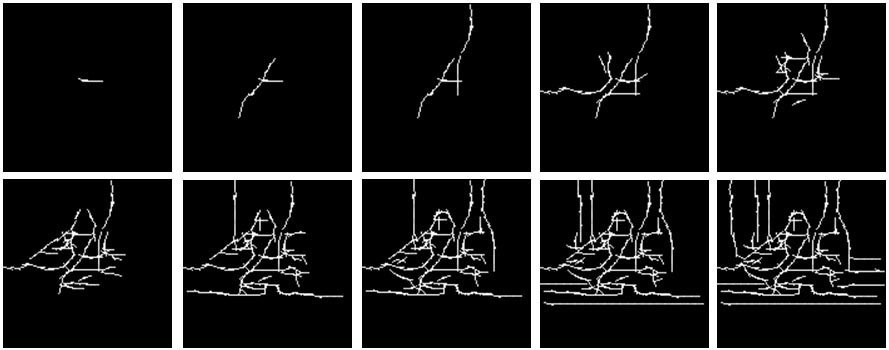


Figure 17: A sequence of routes that have been added successively.

Table 1: Comparison of average precision (AP) on ETHZ Shape classes

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Apple | Bottle | Giraffe | Mug | Swan | Average |
| Our method | 0.875 | 0.963 | **0.859** | **0.906** | **0.972** | **0.915** |
| Lu et al.[52] | 0.844 | 0.641 | 0.617 | 0.643 | 0.798 | 0.709 |
| Felz et al.[53] | 0.891 | 0.950 | 0.608 | 0.721 | 0.391 | 0.712 |
| Maji et al.[54] | 0.869 | 0.724 | 0.742 | 0.806 | 0.716 | 0.771 |
| Srinivasan et al.[33] | 0.845 | 0.916 | 0.787 | 0.888 | 0.922 | 0.872 |
| Wang et al.[55] | 0.866 | **0.975** | 0.832 | 0.843 | 0.828 | 0.869 |
| Lin et al.[56] | **0.909** | 0.898 | 0.811 | 0.893 | 0.964 | 0.895 |

## Object detection test

We test the proposed method on the ETHZ dataset and the INRIA horse dataset for validation. PASCAL criteria are used for the purpose of detection evaluation, i.e., if the detected bounding-box and the ground truth bounding-box intersect over 50% of the union of them, a detection is counted as correct. The proposed model is tested on the state-of-the-art methods [52, 53, 54, 33, 55, 56], we plot the precision-recall (PR) curves in Figure 18 and report the average precision (AP) in Table 1. Table 1 indicates that the performance of our method is either the best or very nearly the best for all image categories.

The INRIA horses dataset is another popular dataset for object detection testing. In this dataset, 50 positive examples and 50 negative examples are randomly selected for training. The final trained model is tested on the remaining images. A few representative shape detection results generated by our method is shown in Figure 19a. Each image shows bounding boxes for one or more detections. Figure 19b presents the averaged false-positives per image (FPPI) vs. recall. Our method is comparable to the-state-of-art method [57, 58, 59, 60, 56].

## Efficiency comparison

The proposed shape-based object detection method requires no expensive training and no high-dimensional feature vectors. We therefore expect our method to exhibit a relatively high efficiency. To this end, we compared the time consumed for our method relative to that of a similar state-of-the-art method. Ma et al. [29] proposed a shape-based object detection method that, like the proposed method, utilizes contour segments as the feature of interest. Based again on the ETHZ dataset, we collected the time costs of the two methods. Figure 20 shows the time comparisons for each category, i.e., giraffe, apple logo, swan, bottle, and mug. In Table 2, the comparison of time cost is shown with detailed statistics. Both of the methods are implemented with MATLAB and run on the same Lenovo PC (with Intel i5 CPU, 4GB RAM). We can see that our method is much less time consuming than the other method, which requires exhaustive searching.

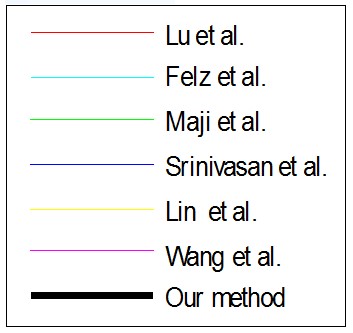
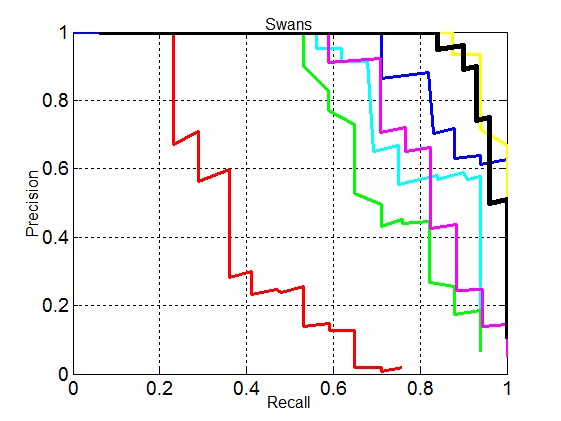
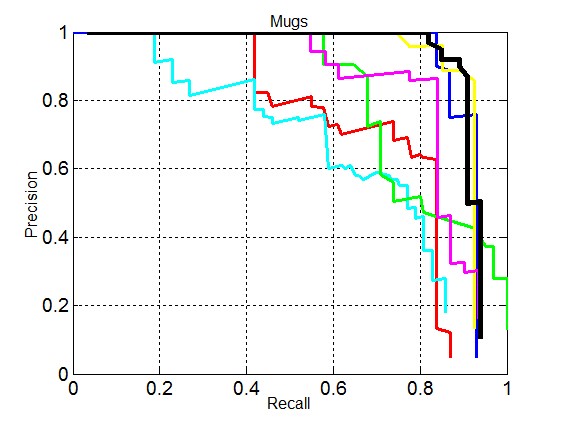
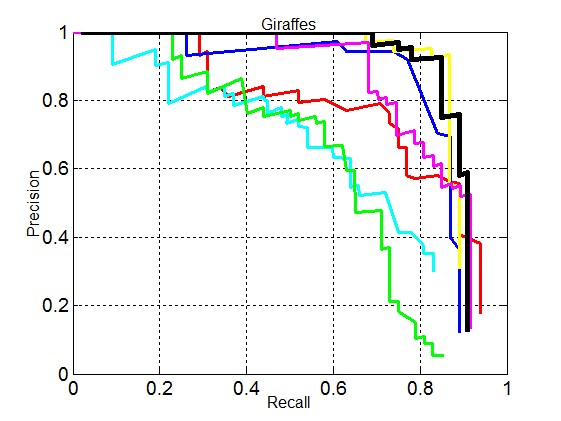
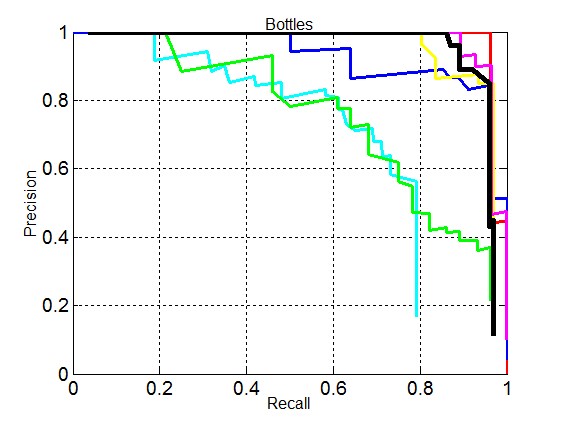
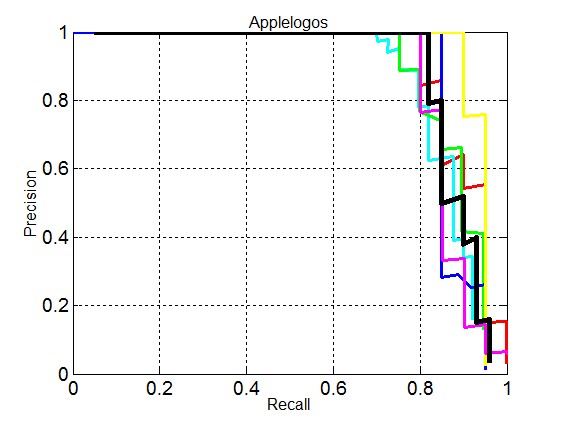
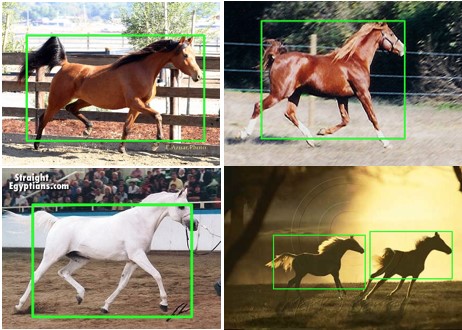
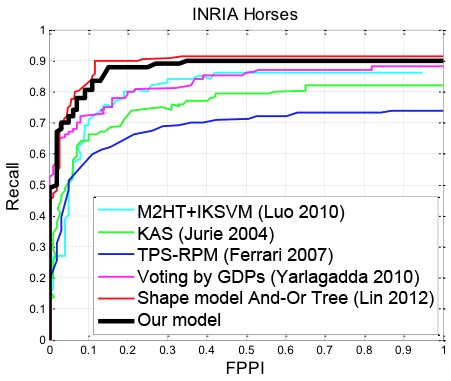


Figure 18: Precision/Recall curves of our method and compared methods on the ETHZ database.



* 1. Representative detection results

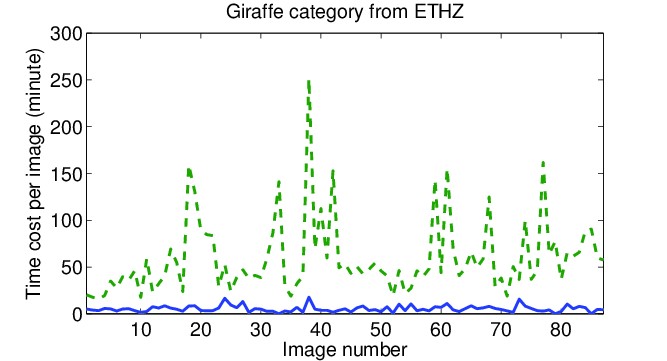
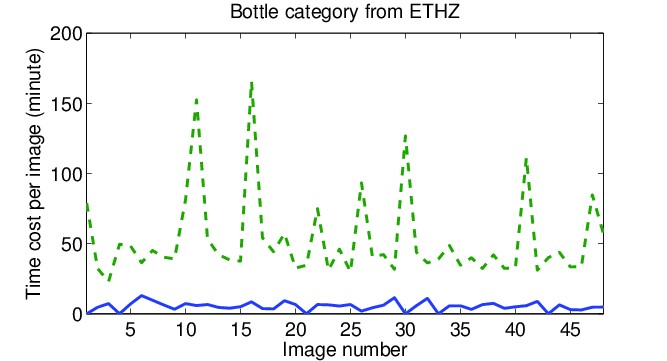
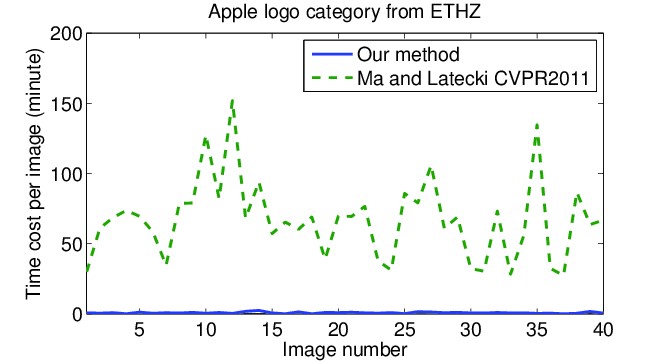


* 1. FPPI-Recall compared with [57, 58, 59, 60, 56]

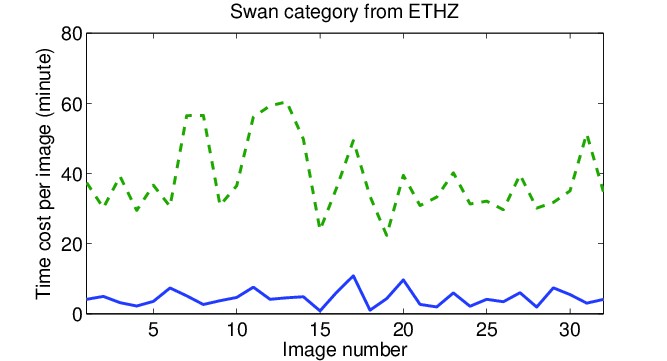
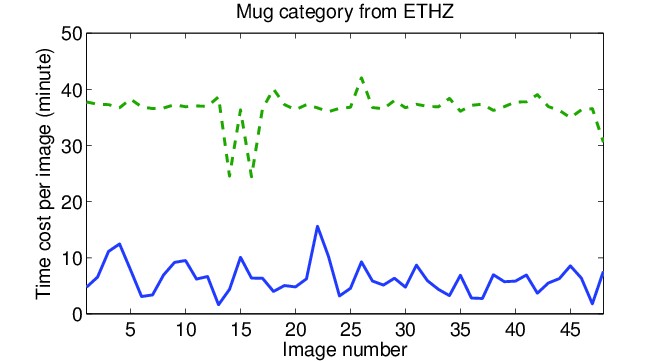
Figure 19: Detection testing for the INRIA horses dataset.

Table 2: Comparison of time cost in minutes per image

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Category | Apple | Bottle | Giraffe | Mug | Swan |
| Our method (mean) | 0.76 | 5.34 | 5.37 | 6.27 | 4.50 |
| Our method (min) | 0.01 | 0.01 | 0.02 | 1.63 | 0.84 |
| Our method (max) | 2.4 | 13.0 | 17.7 | 15.6 | 10.8 |
| Ma et al (mean) | 67.1 | 52.5 | 59.2 | 36.5 | 38.6 |
| Ma et al (min) | 27.3 | 23.1 | 15.9 | 24.4 | 22.3 |
| Ma et al (max) | 151.8 | 165.5 | 251.4 | 42.0 | 60.4 |



(a) (b) (c)



(d) (e)

Figure 20: Time-cost comparison between our method and [29]

## Active processing

The proposed method has a pronounced potential for employing high-level knowledge downwards to actively search an image and collect specific or anticipated geometrical or topological evidence, a process denoted as active processing, or active image analysis. For example, due to some less than ideal preprocessing parameters, a contour segment of an object might be missed at the early phase of analysis, or a crucial contour might be broken into several discontinuous parts, or a contour misjoined with an edge from the background. All these errors can hide valuable evidence and prevent objects from being resolved from the background. However, our method has the capability of correcting these early errors at a later stage through active analysis. Based on a partial match between a template and some real curves, our method can formulate a hypothesis and anticipate what should be the next evidence and where it most likely occurs. Subsequently, a series of intensive operations confined to this designated and limited area are conducted, such as adjusting parameters, executing a new round of preprocessing, and facilitating the emergence of new evidence or reorganizing old evidence and searching within them. This type of processing, under a clear top-down instruction, is active. The most significant advantage of this functionality is that it enables bidirectional, data-driven, and concept-driven processing, and merges them together seamlessly. This functionality greatly improves self-adaptability.

Figure 21 demonstrates how active processing is achieved. In Figure 21a, a suspected route is selected as the beginning of the search because it matches a part of the template well. This segment is re-represented by a route, i.e., a sequence of nodes. We define a state evaluation function for the search process, which is the aforementioned distance value between the current curve and the template. This value, as heuristic information, can be used to guide the search. The present distance value is 20. In Figure 21b, through loosening the threshold defining neighboring nodes, one end node of the red route, 14, has two succeeding nodes, 15 and 19. A tentative, depth-first exploration can return a value of the state evaluation function. The movement towards 15 brings a value of 17.98, and that towards 19 brings a value of 15.54, making the second candidate the better choice, which leads us to Figure 21c. From Figure 21d to 21e the remaining nodes can be evaluated in the same manner, where only the node with the smallest heuristic value can be extended. In particular, Figure 21e indicates that a region-limited preprocessing is executed again under a new canny-detector parameter because the active processing attempts to find a permanently lost edge that bridges the gap between two routes. Finally, as shown in Figure 21f, an almost perfect contour of a bottle is fully detected.

In principle, active processing can be realized under any kind of image representation. However, in practice, an overly fine representation is harmful for efficiency, and an overly abstract representation is harmful for flexibility. The Line-graph and Route-graph mechanism is able to not only integrate discrete pixels to a compound degree, but also preserve sufficient details for intensive analysis. This mechanism not only facilitates searching, but also significantly eases the burden associated with the searching process.

# Conclusion

Figure 22 briefly summaries the overall architecture of the proposed system.

The system has four representation levels: (1) an edge image; (2) a hypercolumn array representing the contour information of the edge image; (3) a graph of neighboring short line segments; and (4) the salient routes of the graph that are candidates for shape matching. The most significant aspect of this process is that the route of a graph provides the proper granularity for the operations of shape-construction and shape-destruction, i.e., the route is sufficiently flexible and concise to be manipulated for the purpose of contour combination. Therefore, the present work provides the following four conclusions.

Perception always is real-time processing, so its efficiency is crucial foran algorithm’s rationality evaluation. In this paper, we achieve the same performance using a relatively simple method, but our method is much more timesaving. From the engineering point of view, the hypercolumn array is a biologically inspired model which helps the design of bionic chips, which allows fine-grained parallel computing with neural circuits and real-time processing could be possible.

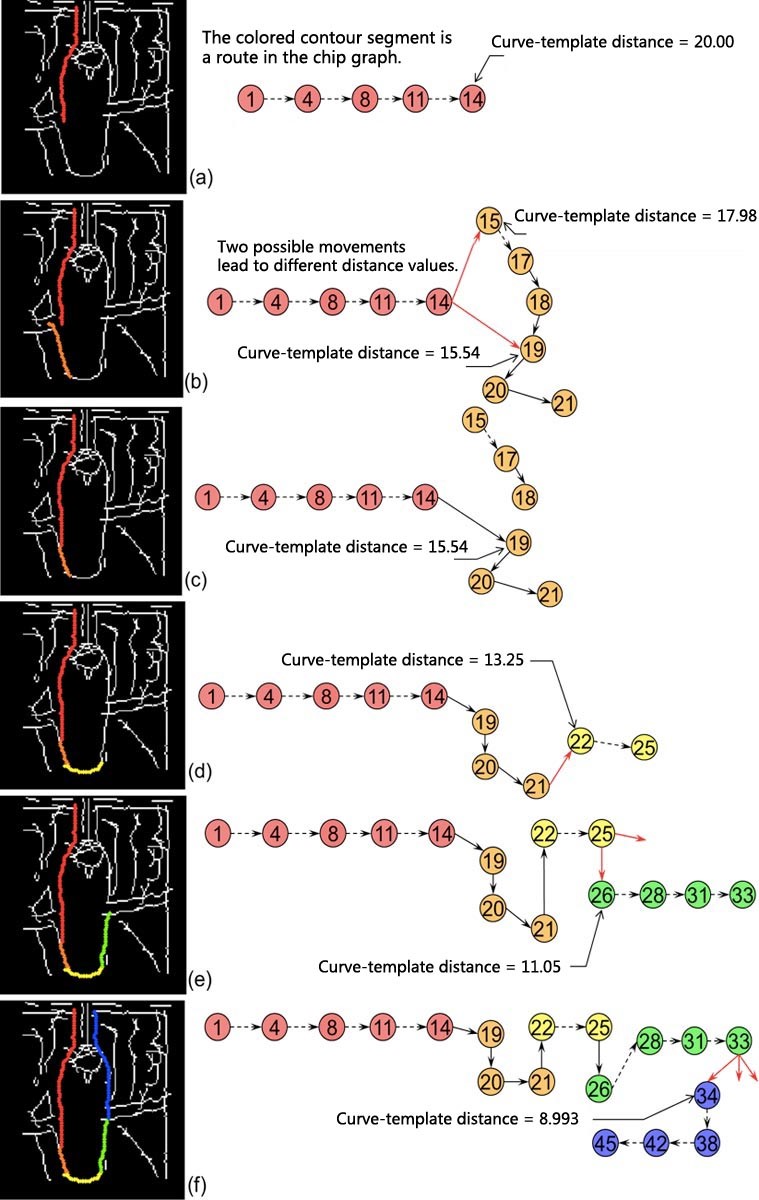


Figure 21: An illustration of active-processing.

A biological visual system especially that of a mammal, has been evolved by nature to be a highly optimized system. Feature-integration theory has been deeply studied in cognitive psychology. Physiological inspirations are indispensable for designing a practical vision system. The hypercolumn array is different from traditional representations such as edge map. It utilizes a limited number of units, or computation resources, to achieve a balance between the complexity of the model and the accuracy of the representation. This optimized design is imitating the result of natural evolution.

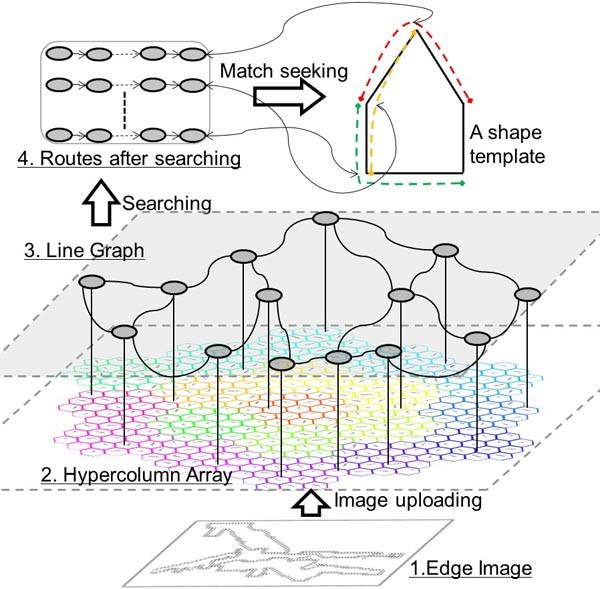


Figure 22: The overall architecture of the proposed system.

1. Task-specific methods, for their diversities, are weak to mature vision theory. A necessary symbol for this theory is it has a unique architecture, whether in representation or in processing. For this sake, choosing geometrical structure and topological relation to represent and choosing shape match to achieve recognition are perhaps on the right track.
2. The training set and appearance-based machine learning strategy for object detection under a natural environment can never exhaust all possible negative cases. Therefore, this strategy is really too expensive for its limited gain in generalization. The strategy of evidence accumulation and inference can avoid these costs.

In this paper, we simulate one of the functions of hypercolumns in primary visual cortex, which is extracting contour features from the visual stimuli. Other functions of the hypercolumns can be added in the near future. These functions involve the color column and the ocular dominance column, which provide color and depth features. With the hypercolumn model, these features can be integrated in the same computation unit. By exploring the degree of sufficiency and necessity of these various features for image processing, we can achieve a balance between the complexity of the model and the overall performance. This helps us to understand how the visual cortex works. It also helps us to find engineering implementation of image understanding algorithms based on multiple clues.

# Acknowledgment

This work was supported by the 973 Program (Project No. 2010CB327900), NSFC project (Project Nos. 61375122 and 30990260). We also thank the anonymous referees for their helpful comments.

# References

1. J. Bickle, M. Bernstein, M. Heatley, C. Worley, and S. Stiehl, “A functional hypothesis for lgn-v1-trn connectivities suggested by computer simulation,” *Journal of Computational Neuroscience*, vol. 6, no. 3, pp. 251–261, 1999.
2. N. Hesam Shariati and A. W. Freeman, “A multi-stage model for fundamental functional properties in primary visual cortex,” *PLoS ONE*, vol. 7, no. 4, p. e34466, 04 2012.
3. H. Nakagama and S. Tanaka, “Self-organization model of cytochrome oxidase blobs and ocular dominance columns in the primary visual cortex,” *Cerebral Cortex*, vol. 14, no. 4, pp. 376–386, 2004.
4. R. Harvey, “The extraction of features and disparities from images by a model based on the neurological organisation of the visual system,” *Vision Research*, vol. 48, no. 11, pp. 1297 – 1306, 2008.
5. B. Curuklu and A. Lansner, “An abstract model of a cortical hypercolumn,” in *Neural Information Processing, 2002. ICONIP ’02. Proceedings of the 9th International Conference on*, vol. 1, Nov 2002, pp. 80–85 vol.1.
6. Z. Wang, L. Shi, H. Wan, and X. Niu, “An information integration model of the primary visual cortex under grating stimulations,” *Biochemical and Biophysical Research Communications*, vol. 413, no. 1, pp. 5 – 9, 2011.
7. Q. Li, D. Lin, and Z. Shi, “A two-layer feedback vision sparse coding model based on multilayer perceptrons,” in *Neural Networks and Brain, 2005. ICNN B ’05. International Conference on*, vol. 3, Oct 2005, pp. nil13–1604.
8. S. Mihalas, Y. Dong, R. von der Heydt, and E. Niebur, “Event-related simulation of neural processing in complex visual scenes,” in *Information Sciences and Systems (CISS), 2011 45th Annual Conference on*, March 2011, pp. 1–6.
9. J. Ramirez-Quintana and M. Chacon-Murguia, “Self-organizing retinotopic maps applied to background modeling for dynamic object segmentation in video sequences,” in *Neural Networks (IJCNN), The 2013 International Joint Conference on*, Aug 2013, pp. 1–8.
10. T. Okamoto, M. Watanabe, K. Aihara, and S. Kondo, “An explanation of contextual modulation by short-range isotropic connections and orientation map geometry in the primary visual cortex,” *Biological Cybernetics*, vol. 91, no. 6, pp. 396–407, 2004.
11. B. D. Willmore, H. Bulstrode, and D. J. Tolhurst, “Contrast normalization contributes to a biologically-plausible model of receptive-field development in primary visual cortex (v1),” *Vision Research*, vol. 54, no. 0, pp. 49 – 60, 2012.
12. J. A. Bednar, “Building a mechanistic model of the development and function of the primary visual cortex,” *Journal of Physiology-Paris*, vol. 106, no. 56, pp. 194 – 211, 2012, new trends in neurogeometrical approaches to the brain and mind problem.
13. J. Law, J. Antolik, and J. Bednar, “Stable and robust development of orientation maps and receptive fields,” *BMC Neuroscience*, vol. 12, no. Suppl 1, p. P10, 2011.
14. C. E. Giacomantonio, M. R. Ibbotson, and G. J. Goodhill, “The influence of restricted orientation rearing on map structure in primary visual cortex,” *NeuroImage*, vol. 52, no. 3, pp. 875 – 883, 2010, computational Models of the Brain.
15. Z. Songnian, Z. Qi, J. Zhen, Y. Guozheng, and Y. Li, “Neural computation of visual imaging based on kronecker product in the primary visual cortex,” *BMC Neuroscience*, vol. 11, no. 1, 2010.
16. X. Yan, A. Khambhati, L. Liu, and T. S. Lee, “Neural dynamics of image representation in the primary visual cortex,” *Journal of Physiology-Paris*, vol. 106, no. 56, pp. 250 – 265, 2012, new trends in neurogeometrical approaches to the brain and mind problem.
17. V. Pi¨ech, W. Li, G. N. Reeke, and C. D. Gilbert, “Network model of topdown influences on local gain and contextual interactions in visual cortex,” *Proceedings of the National Academy of Sciences*, vol. 110, no. 43, pp. E4108–E4117, 2013.
18. S. C, S. DS, and R. G, “Variability in visual cortex size reflects tradeoff between local orientation sensitivity and global orientation modulation.” *Nat Commun*, vol. 4, no. 3, pp. 1345–1346, 2013.
19. J. Canny, “A computational approach to edge detection,” *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, no. 6, pp. 679–698, 1986.
20. P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik, “Contour detection and hierarchical image segmentation,” *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 33, no. 5, pp. 898–916, 2011.
21. R. Moreno, M. A. Garcia, and D. Puig, “Tensor voting for robust color edge detection,” in *Advances in Low-Level Color Image Processing*. Springer, 2014, pp. 279–301.
22. R. G. von Gioi, J. Jakubowicz, J.-M. Morel, and G. Randall, “Lsd: A fast line segment detector with a false detection control,” *IEEE Transactions on Pattern Analysis & Machine Intelligence*, no. 4, pp. 722–732, 2008.
23. G. Borgefors, “Hierarchical chamfer matching: a parametric edge matching algorithm,” *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 10, no. 6, pp. 849 – 865, 1988.
24. J. Shotton, A. Blake, and R. Cipolla, “Contour-based learning for object detection,” in *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on*, 2005, pp. 503 – 510 Vol. 1.
25. A. Thayananthan, B. Stenger, P. H. S. Torr, and R. Cipolla, “Shape context and chamfer matching in cluttered scenes,” in *2013 IEEE Conference on Computer Vision and Pattern Recognition*, 2003, p. 127.
26. A. Opelt, A. Pinz, and A. Zisserman, “A boundary-fragment-model for object detection,” *In Proceedings IEEE Conf. on Computer Vision and Pattern Recognition (CVPR06*, 2006.
27. M. Pavan and M. Pelillo, “Dominant sets and pairwise clustering,” *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 29, no. 1, pp. 167–172, 2007.
28. X. Yang, H. Liu, and L. J. Latecki, “Contour-based object detection as dominant set computation,” *Pattern Recognition*, vol. 45, no. 5, pp. 1927– 1936, 2012.
29. T. Ma and L. Latecki, “From partial shape matching through local deformation to robust global shape similarity for object detection,” in *In CVPR*, 2011, pp. 1441 – 1448.
30. H. Riemenschneider, M. Donoser, and H. Bischof, “Using partial edge contour matches for efficient object category localization,” *Lecture Notes in Computer Science*, pp. 29–42, 2010.
31. H. Liu, L. J. Latecki, and S. Yan, “Robust clustering as ensemble of affinity relations,” *In Neural Info. Proc. Systems (NIPS*, 2010.
32. Q. Zhu, L. Wang, Y. Wu, and J. Shi, “Contour context selection for object detection: A set-to-set contour matching approach.” *Computer Vision ECCV 2008*, 2008.
33. P. Srinivasan, Q. Zhu, and J. Shi, “Many-to-one contour matching for describing and discriminating object shape,” in *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, 2010, pp. 1673 – 1680.
34. R. Fergus, P. Perona, and A. Zisserman, “A visual category filter for google images,” in *Computer Vision-ECCV 2004*. Springer, 2004, pp. 242–256.
35. M. Leordeanu, M. Hebert, and R. Sukthankar, “Beyond local appearance: Category recognition from pairwise interactions of simple features,” in *Computer Vision and Pattern Recognition, 2007. CVPR’07. IEEE Conference on*. IEEE, 2007, pp. 1–8.
36. A. Gupta, J. Shi, and L. S. Davis, “A“shape aware”model for semisupervised learning of objects and its context,” in *Advances in Neural Information Processing Systems*, 2009, pp. 577–584.
37. J. Schlecht and B. Ommer, “Contour-based object detection.” in *BMVC*, 2011, pp. 1–9.
38. V. Ferrari, T. Tuytelaars, and L. Van Gool, “Object detection by contour segment networks,” in *Computer Vision–ECCV 2006*. Springer, 2006, pp. 14–28.
39. H. Wei, Q. Li, and Z. Dong, “Learning and representing object shape through an array of orientation columns,” *Neural Networks and Learning Systems, IEEE Transactions on*, vol. 25, no. 7, pp. 1346–1358, 2014.
40. H. Wei and H. Li, “Shape description and recognition method inspired by the primary visual cortex,” *Cognitive Computation*, vol. 6, no. 2, pp. 164–174, 2014.
41. H. Wei and X.-G. HE, “A hierarchical model based on visual cortex,” *CHINESE JOURNAL OF COMPUTERS-CHINESE EDITION-*, vol. 23, no. 6, pp. 620–628, 2000.
42. K. Schindler and D. Suter, “Object detection by global contour shape,” *Pattern Recognition*, vol. 41, no. 12, pp. 3736–3748, 2008.
43. E. M. Arkin, L. P. Chew, D. P. Huttenlocher, K. Kedem, and J. S. Mitchell, “An efficiently computable metric for comparing polygonal shapes,” *IEEE transactions on pattern analysis and machine intelligence*, vol. 13, no. 3, pp. 209–216, 1991.
44. K. Wang, T. Gasser *et al.*, “Alignment of curves by dynamic time warping,” *The Annals of Statistics*, vol. 25, no. 3, pp. 1251–1276, 1997.
45. B. Liu, S.-X. Xia, F.-R. Meng, and Y. Zhou, “Extreme spectral regression for efficient regularized subspace learning,” *Neurocomputing*, vol. 149, pp. 171–179, 2015.
46. G.-B. Huang, Q.-Y. Zhu, and C.-K. Siew, “Extreme learning machine: theory and applications,” *Neurocomputing*, vol. 70, no. 1, pp. 489–501, 2006.
47. J. Tang, C. Deng, and G.-B. Huang, “Extreme learning machine for multilayer perceptron,” 2015.
48. A. A. Mohammed, R. Minhas, Q. J. Wu, and M. A. Sid-Ahmed, “Human face recognition based on multidimensional pca and extreme learning machine,” *Pattern Recognition*, vol. 44, no. 10, pp. 2588–2597, 2011.
49. G. Huang, S. Song, J. N. Gupta, and C. Wu, “Semi-supervised and unsupervised extreme learning machines,” *Cybernetics, IEEE Transactions on*, vol. 44, no. 12, pp. 2405–2417, 2014.
50. J. C. Russ and R. P. Woods, “The image processing handbook,” *Journal of Computer Assisted Tomography*, vol. 19, no. 6, pp. 979–981, 1995.
51. M. Maire, P. Arbel´aez, C. Fowlkes, and J. Malik, “Using contours to detect and localize junctions in natural images,” in *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*. IEEE, 2008, pp.

1–8.

1. C. Lu, L. J. Latecki, N. Adluru, X. Yang, and H. Ling, “Shape guided contour grouping with particle filters,” in *Computer Vision, 2009 IEEE 12th International Conference on*. IEEE, 2009, pp. 2288–2295.
2. P. F. Felzenszwalb and J. D. Schwartz, “Hierarchical matching of deformable shapes,” in *Computer Vision and Pattern Recognition, 2007. CVPR’07. IEEE Conference on*. IEEE, 2007, pp. 1–8.
3. S. Maji and J. Malik, “Object detection using a max-margin hough transform,” in *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*. IEEE, 2009, pp. 1038–1045.
4. X. Wang, X. Bai, T. Ma, W. Liu, and L. J. Latecki, “Fan shape model for object detection,” in *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*. IEEE, 2012, pp. 151–158.
5. L. Lin, X. Wang, W. Yang, and J. Lai, “Learning contour-fragment-based shape model with and-or tree representation,” in *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*. IEEE, 2012, pp. 135–142.
6. P. Luo, L. Lin, and H. Chao, “Learning shape detector by quantizing curve segments with multiple distance metrics,” in *Computer Vision–ECCV 2010*. Springer, 2010, pp. 342–355.
7. F. Jurie and C. Schmid, “Scale-invariant shape features for recognition of object categories,” in *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on*, vol. 2. IEEE, 2004, pp. II–90.
8. V. Ferrari, F. Jurie, and C. Schmid, “Accurate object detection with deformable shape models learnt from images,” in *Computer Vision and Pattern Recognition, 2007. CVPR’07. IEEE Conference on*. IEEE, 2007, pp. 1–8.
9. P. Yarlagadda, A. Monroy, and B. Ommer, “Voting by grouping dependent parts,” in *Computer Vision–ECCV 2010*. Springer, 2010, pp. 197–210.

1. Hui Wei is with the School of Computer Science, Laboratory of Cognitive Modeling and Algorithms, Brain Science Research Center, Fudan University, Shanghai, China. Corresponding author’s email: weihui@fudan.edu.cn. [↑](#footnote-ref-1)
2. Zheng Dong is with the School of Computer Science, Laboratory of Cognitive Modeling and Algorithms, Fudan University, Shanghai, China. [↑](#footnote-ref-2)
3. Bing Liu is with the School of Computer Science and Technology, China University of Mining and Technology, Xuzhou, Jiangsu Province, China [↑](#footnote-ref-3)