Learning and Representing Object Shape Through an Array of Orientation Columns
Hui Wei, Qiang Li, and Zheng Dong

Abstract—Recognizing an object from its background is always a very challenging task for pattern recognition, especially when the size, pose, or illumination of the object or the background are changing. The most essential method of handling this classical problem is to learn and define the structure of an object using its topological or geometrical features and components. To create a data structure that can describe the spatial relationships of object components formally and join knowledge learning and applying in a seamless loop, a representation platform must be developed. This platform can serve as a shared workspace not only for learning but also for recognition. In this paper, the platform is established by simulating the primary functional modules in the biological primary visual cortex (V1). V1 is located at the middle level of the visual information processing system. As the conjunction of low-level data and high-level knowledge, it performs visual processing for general purposes. Orientation columns in V1 are simulated in our platform, and an array of such columns is designed to represent the orientation features of edges in an image. With this platform, formalized prototypes are designed to represent each typical view of an object and thus the object concept. Data- and concept-driven processing can shift iteratively on this platform. The processes of acquiring knowledge of an object and applying that knowledge coincide with each other perfectly. The experimental results show that our algorithm can learn from a small training set and can recognize the same types of objects in natural background without any preliminary information. This bioinspired representation platform offers a promising prospect for the handling of semantic-concerned problems that need prior knowledge.

Index Terms—Bidirectional processing, neural modeling, object representation, orientation columns.

I. INTRODUCTION

Currently, there are two dominant methods of object recognition, the model-based method and the appearance-based [1] method. The latter is more popular because it requires a fewer preprocessing stages, such as image enhancement, contour detection, segmentation, figure–ground separation, knowledge-based analysis, and topological analysis. This will lighten the burden of detailed image analysis.

II. LEARNING AND REPRESENTING THE SHAPE OF OBJECT ARE CRUCIAL TO IMAGE UNDERSTANDING

Defining the semantics of an object using its structure or shape is more natural than defining the semantics using tags. The topological or geometrical structure of an object can tolerate changes in size, pose, position, color, illumination, and background, which improves invariance recognition. Fig. 1 shows a task assigned to children. In this figure, there are balls and holes. Children are asked to: 1) find the biggest and the smallest ball; 2) find out which two of the balls are of the same size; and 3) find a proper hole for every ball. Learning what a ball is and recognizing its shape are critical to the fulfillment of this task. The appearance-based method is weak here because pixel-level features are insufficient for describing geometrical information.

Once several positive samples of a type of object are available, inductive learning can be used to draw a prototype of an object and apply that knowledge to a new object. This can be achieved through a pattern classification system. In this method, an image is usually described using pixel-level features and sample training is used to acquire some kinds of implicit classification standard. A good generalization can be expected if proper statistical learning algorithms are used. Although it can achieve image classification and image retrieval, this kind of method is rather vague with respect to the semantics of an object. In conventional learning stages, tags or labels are assigned to images to describe their semantics. Therefore, mapping is learned between tags and images. This does answer the question of whether an image is similar to a benchmark, but it cannot answer the question of what the target object is, where it is, or how many objects there are. For example, an image may be tagged a buffalo in a river but tags such as buffalo, river, and bank do not need to be defined explicitly. This leaves the nature of a buffalo, what it is, somewhat uncertain. However, the model-based method is strict with respect to template definition and it is closer to human cognitive processing.

A. Learning and Representing the Shape of Object Are Crucial to Image Understanding

In this method, an image is usually described using pixel-level features and sample training is used to acquire some kinds of implicit classification standard. A good generalization can be expected if proper statistical learning algorithms are used. Although it can achieve image classification and image retrieval, this kind of method is rather vague with respect to the semantics of an object. In conventional learning stages, tags or labels are assigned to images to describe their semantics. Therefore, mapping is learned between tags and images. This does answer the question of whether an image is similar to a benchmark, but it cannot answer the question of what the target object is, where it is, or how many objects there are. For example, an image may be tagged a buffalo in a river but tags such as buffalo, river, and bank do not need to be defined explicitly. This leaves the nature of a buffalo, what it is, somewhat uncertain. However, the model-based method is strict with respect to template definition and it is closer to human cognitive processing.

Fig. 1. Matching task for kindergarten children.
the object. From a psychological point of view, the prototype is an inner representation that must be memorized. For a computer program, formal data structures are used to describe that prototype. One direct method involves choosing some kinds of knowledge representation methods, such as a semantic networks, production rules, frames, and ontology. However, from the point of view of knowledge engineering, a good knowledge representation scheme must be able to smoothly join two levels of knowledge, the level of knowledge acquired from the learning phase and the level of knowledge used in the reasoning phase. The knowledge concerned in this paper pertains to the structural characteristics of an object. These characteristics should be the result of learning but should also be used as an evidence in recognition processes. Although it is possible to define structure-related knowledge manually by knowledge representation tools, such as production rules, however, it breaks the natural transition from learning to recognition because the form of knowledge acquired during the learning phase could be out of joint with the form of knowledge used in the reasoning phase. The knowledge concerned in this paper pertains to the structural characteristics of an object. These characteristics should be the result of learning but should also be used as an evidence in recognition processes. Although it is possible to define structure-related knowledge manually by knowledge representation tools, such as production rules, however, it breaks the natural transition from learning to recognition because the form of knowledge acquired during the learning phase could be out of joint with the form that they are required in recognition. Therefore, a representative form that runs through the whole process of perception in a natural and uniform way should be developed.

The goal of this paper is to determine the structure of an object, represent it in an efficient form, and use this prototype or template to recognize other samples.

B. Related Work

This paper concerns two types of studies, 1) object representation and recognition and 2) computational simulation of primary visual cortex (V1).

With respect to object definition, four previous studies used chain codes to represent objects’ contour [2]–[5]. The major limitation of this approach is that the chain code is too small in scale to reflect any macrofeature, such as the slope of a line. This involves considerable data storage costs for contour representation. Another three studies used spline curves to fit objects’ contours [6]–[8]. The major limitation of this method is that the accuracy with which the shape is represented is highly dependent on point-set and coordinate system selection. These methods all have direct and marked effects on fitting computation. Some previous studies used polygon approximation for shape representation [9]–[11]. The major limitation of this approach is that it is posture and size sensitive. This can affect the forming of polygons. In [12]–[15], scale-space features were used to represent original images. The major limitation of this method is that color, brightness, and noise may make the object difficult to describe and make it impossible to produce accurate descriptors for parts of the object. In [16]–[18], the use of global geometrical features, such as area, perimeter, major axis direction, eccentricity, compactness, and convexity, were described. However, these measures cannot describe the shape of an object accurately. This is because shape is not necessarily related to any of these measures. Three other studies describe experiments based on Hu-moment, Fourier, wavelet, or geometric spectrum descriptors [19]–[22]. The major limitations of these methods are color, brightness, and noise sensitivity. These methods are limited in intensive position, posture, and size analysis because the parameters, such as moment values or Fourier coefficients, usually obscure this information. Some previous studies used a template [23]–[25]. This strategy is consistent with psychological theory [26]. It can actually reflect human cognitive processing. This paper also follows this theory.

With respect to V1 modeling, many simulation-related works were done. In [27]–[31], biological cortical maps were used. In [32]–[40], computational cortical maps were used. This map is also a concern in this paper. The map’s efficiency with respect to image representation is addressed here. In [41]–[43], hardware chips were used to simulate V1. In [44], another supercomputing-based method was proposed. In [45]–[49], some V1 mechanisms were simulated and used for invariance recognition for some special purposes. However, these simulations were somewhat crude and some basic physiological constraints were ignored. In [50]–[52], the modeling of functional columns of V1 for neural biological purposes were described. Six more also simulated functional columns of V1 for task-specific applications [53]–[59]. The purpose of this paper is somewhat different from those of the previous studies. The present model for V1 has more computational reliability, on the other hand, this model can be used in a more general way for image representation. Not all of these diverse vision-dependent tasks have been diverged even at the in V1 stage. The intended representation schema inspired by V1 should also be multipurpose.

The limitations of the previous works highlight the value of the current model.

C. Paper Organization

Sections I-A and I-B summarize the motivation behind this paper. Section II discusses single orientation column modeling after a brief introduction of biological ice-cube model. Then, an artificial column array was designed for edged image representation. Here, hexagons in two scales were used to construct the physical model of the column array. The output of column array is vectorized lines. Section III suggests a line-context descriptor and discusses how to use descriptors to represent an object. This process turns raw image data into structured vector data. Section IV details how to learn features from multiple samples and how to form different prototypes of a type of object. Top–down processing is crucial to object recognition. Section V addresses how bidirectional processing works based on this new representation schema. Section VI describes the present set of experiments, including a representation stability experiment, an object detection statistical experiment, and active processing experiment. Finally, Section VII describes the present conclusion and prospects for future work.

II. ORIENTATION COLUMNS MODELING

Biological visual systems, especially the visual systems of higher mammals, have been profoundly optimized by natural evolution. The tradeoff between system complexity and function is ideal. The paper involves simulating the actions of the primary visual cortex (V1) because of its hub role in visual information processing. All information from the lower retina
comes here and all information that is eventually sent to the higher cortex starts here. The information stored in the primary cortex is sufficient for upcoming advanced processing.

A. Neurophysiological Mechanism of Orientation Column

In the V1 area of the striate cortex, many cells respond strongly to special orientations [39], [60]. These are called simple cells. This kind of directional selectivity was studied by Hubel and Wiesel [61]. They won the 1981 Nobel Prize. They suggested the concept of an orientation column in V1, in which the orientation selection of simple cells in a column varies systematically and adjacent cells share approximate orientation preferences [40]. Hubel and Wiesel proposed that this column should be the basic unit of functional organization. Fig. 2 shows columns redrawn in terms of the results in [62]. This shows the ice-cube model of the cortex. It illustrates how the cortex is divided into two kinds of slabs, one set for ocular dominance (L here is left and R is right) and one set for orientation [63].

B. Hint for Computational Model Design

From the perspective of computer programming, the ice-cube model discussed in the previous section is an ideal data structure for recording orientations. One column is considered the primary functional unit, and an array of columns can serve as a representation platform. This is similar to the duplicate micromodules in V1 that can be found during visual inspection of its anatomical structure. The edge detection algorithm puts out a boundary image, and boundaries serve as lines, also called orientations in some studies. The ice-cube model inspired us to design a similar array to record the orientation distribution of an image and we believe that contour information must be included in this distribution. The significance of this artificial array is not limited to the modeling of simple cells in V1. It can also be used to create new image-based representation schemes using contour information. This is completely different from other representation schemes, such as histogram-based, Fourier-transform-based, wavelet-based, Scale Invariant Feature Transform (SIFT)-descriptor-based, and basis-function-based schemes, because orientation is a higher order statistical feature and can reflect more geometrical attributes in a local region than other features can.

C. Modeling of Orientation Columns

Let us begin with the design of a single unit that simulates the orientation column in V1. Each orientation column is in charge of a very small part of the stimulus, called the receptive field of the column. Any orientation may vary from 0° to 180°.

This closed interval is divided into subintervals where every span is about 10°. In this way, the orientations that are grouped into a common subinterval have approximately the same slope. In V1, many small chips (simple cells) have evolved to stand for these orientations one by one. That is, once an orientation occurs, a corresponding chip fires. This is called the orientation selectivity of the simple cell. An orientation column assigns a chip to represent each possible subinterval of each slope; these chips are called orientation chips. All of the orientation chips that share a common receptive field and vary their sensitive slopes from 0° to 180° make up a column.

According to the preceding principle, an orientation column can be designed in the form of a large hexagon composed of 19 smaller hexagons. Each small hexagon is an orientation chip, and each orientation chip has a receptive fields inherited from its column. Three neighboring columns are shown in Fig. 3 (right). Their receptive fields overlap somewhat, and they share some low-level ganglion and Lateral Geniculate Nucleus cells. Each of these cells obtains color and grayscale information from a pixelated image.

A hexagon was chosen as the basic module unit because 1) the slope span of every two neighboring orientation chips is \((180/19)°\), which is sufficiently fine and precise, and 2) 19 small hexagons can cover a big hexagon completely. This makes the array of orientation chips compact, regular, and orderly. Constructing a dense array of small hexagons is a strict work. This is followed by confirming that the coverage of the visual field is continuous. This makes it easier to reorganize the 19 small hexagons with the same receptive field into an orientation column.

To guarantee that no information has been missed, a pair of neighboring orientation columns with overlapping receptive fields was established. In this paper, every column has a receptive field of \(9 \times 9\) pixels, which is shared by all chips in the column. The organization of the receptive fields is shown in Fig. 4. On the right are seven orientation columns and on the left are their overlapping receptive fields. The Roman numerals show a one-to-one relationship between a given column and its own receptive field.

Fig. 4 shows a conceptual model. The hexagon array is not conventional. It is usually preferable to use orthogonal grids because they are easy to store in space. Fig. 5 shows the storage settlement of these orientation columns with respect to memory allocation. This kind of transformation serves the data structure design.

More implementation details can also be seen in [64].
Fig. 4. Organization of overlapping receptive fields. (a) Overlapping RFs. (b) Orientation columns.

Fig. 5. Storage model of neighboring orientation columns.

Fig. 6. Orientation representation array initialized by training. (a) Part of an orientation array. (b) Examples of pinwheel locations.

D. Array of Orientation Columns

The purpose of designing a V1-inspired orientation column is to use it to extract orientations within a small region. Rather, it is a kind of feature descriptor that can be used to exhaust a limited space of orientation. For an extended area, an array of orientation columns can be expected to act as infrastructure for image processing. The principle of how to use multiple column units to construct an array, just like splicing tiles on a floor, is shown in Figs. 3 and 4. The manner in which an orientation column array works is also addressed.

First, the array must respond to the orientation stimuli correctly. Using Kohonen’s self-organizing mapping (SOM) algorithm, all orientation chips were trained using many real lines [64]. The advantages of this strategy are 1) the SOM algorithm has strong brain-like computation background and 2) the mode of its training samples coincides with the prospective mode of its stimuli in a real image. The validity of using SOM to simulate cortical map has also been confirmed in some pure neurobiological studies [65].

Fig. 6(a) shows a part of an array after training. Here, every chip has developed its own sensitive orientation. The receptive field partially sharing effect can be seen at the pinwheel locations marked by red circles, as in Fig. 6(b). This phenomenon is an important characteristic in the biological V1 cortex [66], [67]. It shows that the distribution of feature detectors tends to be continuous in any local region. This guarantees that no representation will be missed, regardless of where the object appears. Fig. 7(a) shows a map of orientation sensitivity obtained in an electrophysiological experiment [60]. Fig. 7(b) and (c) shows the results of the present simulation. As in Fig. 7(a), they show the orientation selections of chips in false color. Regarding the mode of organization of the functional units, the artificial results can be shown to be similar to natural results. They are similar in the following respects: 1) no types of orientation selection are dominant or densely distributed; 2) all orientation selections are equally balanced; and 3) in any continuous block, all orientation selections are well proportioned. In this way, it can be expected that, in a computational sense, if the natural cortical map can record orientation information efficiently and reliably, then the artificial map can as well.

Fig. 8 shows an example of how an image is represented. On the left is a stimulus that includes lines. On the right is an orientation column array, wherein each chip has a special sensitive direction. The three circles show three columns and their corresponding receptive fields are represented by the three dotted-line rectangles on the left. The small black hexagons represent chips that are activated by bars.

Boundary information is very important for any shape-based algorithm. Edge output by algorithms like the Canny operator can be considered as fine bars. Then, a column array can be used to extract orientation information from a scene. The current representation through an orientation column array is much more condensed than a representation through pixels because orientations and lines are more informative than pixels. Local topological information of the images is well reserved. The orientation is a higher order statistical feature in that it is more directly concerned with the shape of an object. These are used to describe the geometric and structural characteristics of an object.
III. TYPE OF LINE-CONTEXT DESCRIPTOR

The process in which an edged image is represented in dozens of active orientation chips, is like using letters (chips) to describe an object. It may be possible to upgrade them to words or sentences. This means that these active orientation chips must be organized.

A. Revised Shape-Context Descriptor

Although orientation chips have been shown to be able to integrate information to some extent in local areas, they are still too fine in scale to use over large areas. Here, the famous shape-context descriptor is revised [68]–[70]. A new descriptor is designed for the description of the distribution of active chips. It is here called a line-context descriptor.

Fig. 9 shows an array of orientation columns in which some of the chips, represented by black hexagons, have been activated by a stimulus. On any active chip, a polar coordinate system whose starting axis is coincident with this chip’s sensitive slope can be constructed. The positive direction coincides with the direction of that column sorted by a spiral coordinate system, as shown in Fig. 9(b). As in the shape-context method, the whole area of the polar coordinate system can be divided into 4 ~ 6 rings. Each ring is evenly divided into 6 ~ 8 sectors. Every sector is a bin in which the active chips can be added up as follows.

An important technical matter here is determining which bin a column belongs to while the polar coordinate system is changing with the center chip’s position and slope. Fig. 10 shows a solution. Suppose the center of the polar coordinate system is column \( A(x_0, y_0) \), and column \( B(x_3, y_3) \)'s ownership is to be determined. \( \alpha \) is the sensitive orientation of the center chip, and bin shows the scope of a bin after the coordinate system is rotated. \( r \) is the length of one side of a hexagon. Then, the ring index \( i_{\text{ring}} \) and the sector index \( j_{\text{sector}} \) to which column \( B \) belongs are calculated using the following equations:

\[
i_{\text{ring}} = \frac{\sqrt{(x_1 - x_0)^2 + (y_1 - y_0)^2}}{\sqrt{(r \times \frac{\pi}{19})^2 + (r \times 7 \times \frac{\pi}{19})^2}}
\]

\[
j_{\text{sector}} = \frac{\|\text{bin}\|}{2\pi} \left( \left\lfloor \frac{\sin^{-1}\left(\frac{x_1 - x_0}{\sqrt{(x_1 - x_0)^2 + (y_1 - y_0)^2}} - \alpha\right)}{\frac{\pi}{19}} \right\rfloor \right) + 1
\]

Like the shape-context algorithm, this algorithm also collects statistics from these bins. Suppose the polar coordinate system has four concentric rings and each of them is divided into eight divisions [Fig. 9(a)]. We center it at an active chip \( C_0 \) and then a line-context descriptor \((\text{LD})\) is a matrix

\[
\text{LD}(C_0) = \begin{pmatrix} b_{11} & \cdots & b_{1n} \\ \vdots & \ddots & \vdots \\ b_{mn} & \cdots & b_{nn} \end{pmatrix}_{4 \times 8}^{(C_0)}
\]

where \( b_{ij} \) is a 1-D histogram which represents the distribution of relative orientations of active chips in bin \( ij \). The relative orientation of an active chip is the difference of its own sensitive orientation and that of the center chip \( C_0 \). The histogram is defined as a vector \( b_{ij} = (h_1, h_2, \ldots, h_{19})^T \), where

\[
h_k = \begin{cases} C | C \in \text{Active}_{\text{bin}ij} & \text{and} & \frac{(k - 1)\pi}{19} \leq |O_C - O_{C_0}| \leq \frac{k\pi}{19} \end{cases}
\]

for \( k = 1, 2, \ldots, 19 \). Active \( \text{bin}ij \) is the set of active chips in bin \( ij \) and \( |O_C - O_{C_0}| \) is the orientation difference between a chip \( C \) and the center chip \( C_0 \). Therefore, what we finally obtain is a matrix like Table I.

For example, the number 3 in the second row and the third column means that in bin \( 1,2 \) (i.e., sector 2 of ring 1), there are three active chips whose orientation difference relative to the center chip is less than \( 3\pi/19 \) and greater than \( 2\pi/19 \).

The line-context descriptor is a 3-D matrix, which presents a kind of feature that describes how those surrounding lines distribute around a center one. Fig. 11 shows a visualization...
of a line context. One active chip (in black) was selected and its line context was listed at right by a 3-D bar graph. The x-, y-, and z-axes show bin level, interval of relative orientation, and quantity of active chips, respectively.

This descriptor can be local or global, depending on the scope of the polar coordinate system being extended.

IV. MULTIPROTOTYPE REPRESENTATION OF AN OBJECT

The views of an object under different postures are always different. In this way, multiple prototypes with different structural characteristics can be formed for those typical views. The theory of multiple prototypes is helpful for recognition of invariance, so we want our program to learn some prototypes first, and then apply them in future recognition.

A. Standard Samples for Prototype Learning

The learning process here is to have our program grasp those essential and structural characteristics of an object in different postures. For accurate learning, standard images were selected for use as learning samples. For example, Fig. 12 shows some clean pictures from the ETH-80 image database. They can be considered typical images of this kind of car. They were chosen for the construction of prototypes through learning.

These pictures were subjected to edge detection and uploaded to the orientation column array for representation.

B. Inductive Learning of Prototype

Once an image has been processed by the orientation column array, a honeycomb-like output becomes available. Then, as arranged, the line-context algorithm is used upon those activated orientation chips and a number of line-context descriptors will come into being. This means a learning sample has been decomposed into a set of descriptors. If, from the point of view of relational database, the descriptors are considered features, then any set of descriptors can be treated as a sheet of entries or a Microsoft Excel table. In this way, all processed learning samples can be stored in the form of a database sheet. This fits the goal of this paper exactly. Its consequent benefit is immediate and significant because it can represent multiple prototypes in a table now, and its form is very regular and structured. A pixel image is usually considered unstructured raw data, but it can here be converted to a structured item.

To produce a better structured sheet, the degrees of similarity between different samples must be established. Only by doing so, shared structural features can be identified. Fig. 13 shows the reason for this. It shows six learning samples. Circled numbers show the center position of a line-context descriptor. Pairs of descriptors shown in the same colors should match each other. Only those matched descriptors can be identified as features because they might occur repeatedly in more than one instance, but they do not do so by accident. Further explanation of this figure is as follows.

The degree to which the descriptors match one another must be determined. A measure of similarity between two descriptors can be defined. It involves measuring the similarity between two corresponding bins and then adding the weighted values of all bin-to-bin similarities together to produce a value representing descriptor-to-descriptor similarity. The similarity between two bins is defined as follows:

\[
S_{\text{bin}}(b, b') = 1 - \frac{1}{19} \sum_{k=1}^{19} \frac{|h_k - h'_k|}{\max(h_k - h'_k)}
\]  

(2)

where each bin is a histogram over 19 intervals, \(b = (h_1, h_2, \ldots, h_{19})^T\), and \(b' = (h'_1, h'_2, \ldots, h'_{19})^T\). The similarity between two line-context descriptors is the weighted sum of similarity of corresponding bins

\[
S_{\text{descriptor}}(LD, LD') = \sum_{i=1}^{4} \sum_{j=1}^{8} w_{ij} \times S_{\text{bin}}(b_{ij}, b'_{ij})
\]  

(3)

where \(w_{ij}\) is the element of a predefined weight matrix and \(b_{ij}\) and \(b'_{ij}\) are the elements of the descriptor matrices \(LD\) and \(LD'\), as defined by (1) in Section III-A.
Algorithm 1 Line-Context Descriptor Alignment(pool)

Require: pool is a set of line-context descriptors \{LD_{i,j}\}, where \(i = 1, \ldots, M\) is the index of samples and \(j = 1, \ldots, N_i\) is the index of line-context descriptors of the \(i\)th sample.

Ensure: Each line-context descriptor in pool is assigned with a class label.

class ← 0

for \(LD_{i,j} \in pool\)
    if \(LD_{i,j}.\text{class label} \) is not assigned then
        class ← class + 1
        \(LD_{i,j}.\text{class label} \) ← class
    end if
    for \(LD_{i',j'} \in pool\), where \(i' > i\)
        if \(S_{\text{descriptor}}(LD_{i,j}, LD_{i',j'}) \geq 0.7\) then
            \(LD_{i',j'}.\text{class label} \) ← class
        end if
    end for
end for

Suppose that there are \(M\) samples available for learning an object, and each sample is a typical prototype. After orientation column array processing and line-context representation, a set \(\{LD_{s,j}\} 1 \leq s \leq M\) is produced, where \(N_i\) is the number of descriptors obtained from the \(s\)th sample and \(LD_{s,j}\) represents the \(j\)th line-context descriptor of the \(s\)th sample. After expanding this set into lists, its data structure will look like this:

\[
\text{pool} = \left\{ \begin{array}{c}
sample_1 : LD_{1,1}, LD_{1,2}, \ldots, LD_{1,N_1} \\
sample_2 : LD_{2,1}, LD_{2,2}, \ldots, LD_{2,N_2} \\
\vdots \\
sample_M : LD_{M,1}, LD_{M,2}, \ldots, LD_{M,N_M} \end{array} \right\}.
\]

This is called a pool of line-context descriptors.

Therefore, based on the similarity defined in (3), descriptors in different samples that are actually mapping to each other can be identified. This allows the user to find out which descriptors occur frequently. Algorithm 1 is used for this purpose.

After the pool of line-context descriptors are processed by the alignment algorithm, all line-context descriptors in pool can be classified into a dozen of subsets. Each subset contains multiple similar line-context descriptors, which come from different samples. If a subset is considered a feature, then its dimension means that this feature can be observed in more than one prototype of this object. If these features are arranged in a table, then a well-structured sheet can be formed. Finally, normal images can be converted to a Boolean sheet attributed by geometrical or topological features. An example is shown in Table II.

Refer to Fig. 13. After descriptor clustering, the matched descriptors are marked by numbers with the same value and color. Each descriptor’s geometrical meaning is obvious.

This produces important progress in transforming a geometrical problem into a logical one in that this well-formed representation opens doors to many other mature machine learning algorithms, all of which require structured data.

C. Applying a Decision Tree Algorithm

The data presented in Table II strongly suggest that a decision tree algorithm would be suitable. This table can be used to train and build a decision tree, which is not only the classifier for new samples but also an explicit knowledge representation form in that it reveals the logical associations between multiple prototypes and observable features (or evidence). Fig. 14 shows one of the branches of a real decision tree, which is produced by four types of objects selected from the COIL-20 data set and based on WEKAs J48 decision tree tool. This algorithm is well developed, so the details regarding that algorithm are omitted here.

It must be emphasized that the decision tree algorithm itself is not our goal because it has been carefully studied. What we are interested in is the course of transforming a bitmap image into a Boolean sheet and making a decision tree algorithm run on such a symbolic level. Some reasoning methods using symbolic representation are worth further investigation.

V. CONCEPT-DRIVEN PROCESSING

Cognitive psychology dictates that perception is a dual-directional and iterative process. Normally, the data-driven process is well studied, but concept-driven processes are not. The key to concept-driven processing lies in two points. The first is a proper high-level representation form for concepts, and the second is a seamless route for feeding back top-down controls.

Fig. 15 shows the present solution. Once a concept has been learned, according to the prototype theory given in the previous section, it must be defined using multiple geometric features. In actuality, these features are descriptors (top-right corner of Fig. 15). Each descriptor then records a special distribution of short lines, which are located at some specific positions. In this way, an active descriptor can conversely project its expectation to a column array and trigger a comparison between the actual stimulus and the expectation. Then, some pixel-concerned
Algorithm 2 Bidirectional Recognition($D$)

Require: $D$ is a set of line-context descriptors to recognize; conc is an array of concepts; each concept is a symbolic representation of an object class; the symbolic representation could be a decision tree as described above but not necessarily restricted to it; threshold is a threshold value for matching ratio and its default value is 75%

Ensure: return value is the concept, which is recognized from $D$

for $c \in \text{conc}$ do
    ratio ← Matching-Ratio($D$, $c$)
    if ratio > threshold then
        return $c$
    end if
    Rank concept $c$ with ratio
end for

for $c \in \text{conc}$ by decreasing order of ranks do
    repeat
        Project unmatched descriptors in concept $c$ downward
        Reorganize raw data with new hypothesis
        Update the set of descriptors $D$
        ratio ← Matching-Ratio($D$, $c$)
    until ratio > threshold or maximum iterations exceeded
    if ratio > threshold then
        return $c$
    end if
end for

return null {showing a failure of recognition}

manipulations, such as researching, reselecting, rediscovering, and reorganizing pixels, can be performed (top–down arrow in the middle of Fig. 15). Each bin of a descriptor has a limited area within the pixel image for which it is responsible. For the purpose of confirming certain hypotheses, it can accurately project a search request onto this area (bottom of Fig. 15). Algorithm 2 describes this bidirectional recognition.

VI. EXPERIMENT

To prove the feasibility of the current method, a series of follow-up experiments were conducted.

A. About the Representation Diversity Between Objects

The aim of this experiment is to prove that object representation based on line-context descriptors possesses sufficient diversity to differentiate each object. We randomly select five classes of objects from the COIL-20 data set. Each class includes five samples. The five samples are similar to each other and thus belong to the same prototype. The reason for such selection of samples is based on the fact that different views of an object have generally very different geometrical characteristics, which implies that a stable representation can only be reached under a single view (or prototype). Different views are represented by different prototypes. They are first represented with descriptor vectors and then the Euclidean distance between any two representation vectors can be calculated. Fig. 16 shows these samples and a curved surface that records the distances between vectors.

Five areas, labeled A, B, C, D, and E, are shown in Fig. 16. Comparisons can be made among objects in the same class, where the distances are small. Conversely, other areas show comparisons between objects in different classes, so the large distances are obvious. It can therefore be concluded that this representation schema provides not only intraprototype similarity but also interprototype distinguishability and can be applied to object recognition.

B. About Recognition

After an image is transformed into a structured vector, almost all classification algorithms are suitable. A decision tree algorithm was selected here, because 1) it has a simple and clear architecture to realize decision making; 2) it allows convenient extraction of explicit knowledge for object recognition; 3) it is scalable and incremental; and 4) it is suitable for symbolic reasoning and reasoning with incomplete information. All of these reasons are attractive for object recognition. This experiment, presented here, is somewhat different from conventional methods (splitting a fixed data set into a training part and a testing part), because the samples were not confined to a limited data set. Some standard prototypes were obtained from a small model, and then the decision tree was trained using these prototypes. After this, the tree was tested using some real instances in an actual scene. Fig. 17 shows this process. Fig. 17(a) shows a plastic F117 model. Its contour...
can be used as a learning sample. Fig. 17(b) and (c) shows two actual pictures, which include multiple F117s in different sizes. A flexible window was used to search each picture and capture blocks for line-context processing. This was then uploaded to a decision tree. Each red dot box represents a sample and the value attached to each box is the accumulation of feature difference degrees while comparing it with decision tree nodes. We can see that real objects have significantly smaller values.

C. Comparison Using ETH80 Data Set

Statistical comparison between different algorithms is very necessary, but it requires a common image data set. The data sets, such as COIL-20, ETH80, MPEG7, and the Princeton 3-D database, include many samples from the same type of object, but all images have almost the same background. A clean, monotonic background can counteract the need for shape definition. As a tradeoff, the ETH80 data set is still chosen in our experiment because its colorful images benefit appearance-based methods. Here, some of the images were reconstructed. This involves cutting an original image into small blocks, rotating some randomly selected blocks, and then reshuffling and rearranging all of the blocks into a new image. Fig. 18(a) shows an example, which can be used to determine whether a recognition algorithm is local physical feature dominant or not. We then established a database for algorithm testing. First, we designated a car as the recognition target, and all car images come from ETH80. Some 10% of the images in the present database are of cars. Second, the remaining 90% of the images were randomly selected from ETH80. Third, 50% of the samples from the present database were randomly selected for reconstruction. For comparison, the widely accepted SIFT descriptor [71] and shape-context algorithm [70] were selected. The former of these is bioinspired and orientation concerned and the latter is a famous contour-oriented method. Fig. 18(b) and (c) shows correspondences found between two car images using SIFT feature matching and shape-context matching. All three approaches were evaluated over our data set. The data set was classified into car images and noncar images with these approaches. Classification errors (false positives and false negatives) were used as a measure of performance. Finally, precision–recall curves were drawn [Fig. 18(d)]. Our results are much better than other two approaches.

One explanation for the excellent performance of the present algorithm is its improvement in representation over existing methods. For example, 1) the SIFT representation considers much more unimportant background information; 2) the shape-context representation is more sensitive to the number of pixel points rather than their correlations; and 3) our representation is based on a line vector, which contains more geometrical information than SIFT descriptor or the shape-context’s histogram of the number of pixels.

D. Comparison Using ETHZ Data Set

The ETHZ [72] data set is another widely used and well-studied database used for object detection. In this data set, there are five types of objects, i.e., the Apple logo, bottle, giraffe, mug, and swan. In each category, there are more
Fig. 19. Statistical comparison based on the ETHZ data set. (a) to (e) show the results over 5 object categories respectively.

Fig. 20. Top–down active processing. (a) Original model of a Ford Transit van, and its contour image. Red rectangles: specific areas, including dominant features such as wheels and windows. They are good for defining local line-context descriptors. (b) Twenty areas used to define line-context descriptors. (c) Real scene including two Ford Transit vans. They are different from the van in (a). (d) Three local areas are validated and shown to match predefined descriptors. (e) Nine descriptors are matched. (f) Only the front half of the second van is visible. According to those matched descriptors, the rear half of it has been rebuilt successfully using the learned concept Transit van and the rest of the descriptors.

Commonly used false positive per image (FPPI)–detection rate (DR) curve is also used here. The $x$-axis is FPPI and the $y$-axis is DR. Detection was considered to have failed if more than half of the candidate results in a given image were false positives, because DR can generally increase up to one as long as FPPI is unrestricted. For this reason, the proportion of false positives per image was required to not exceed 0.5. Fig. 19 shows the comparisons. In a condition of $\text{FPPI} \leq 0.5$, Fig. 19(a)–(e), shows the performance of our algorithm to
approach that of the best algorithm shown here. Surveying the results of recognition experiments in this and previous sections, we conclude that the proposed object representation method can improve the performance of object recognition.

E. Active Processing

Bidirectional processing is an attractive challenge that has troubled pattern recognition studies for a long time. The difficulty exists in the top–down direction. Specifically, it is difficult to apply high-level knowledge downward to low-level stimuli. A successful example using our algorithm is shown here. According to algorithm 5.1, the learned line-context descriptors can be projected upon an image using a candidate object. An active search and comparison procedure was used to verify and validate the hypothesis. Fig. 20 shows this top–down processing. The present algorithm was first used to determine which descriptors in a given real scene matched the learned descriptors. Then, according to the hypothesis produced by those matched descriptors, the locations and identities of the descriptors were predicted.

From this example, it can be seen that 1) the present method is sufficiently generalized, as shown by the fact that it can adapt two totally different scenes, a learning model in one and a recognized sample in the other; 2) the ability of generalization is not gained by repetitious training on different cases but rather by learning intrinsic structural features; 3) this is a kind of method of intensive analysis, which can be used to position an object’s components accurately; and 4) some invisible or hidden components of an object can be expected according to known evidence, and this is enabled by active processing.

In summary, the color, size, and environmental tolerances of this algorithm are good. If the diversity of prototypes is rich enough, then posture invariance can be achieved easily.

VII. Conclusion

In this paper, only simple mathematical tools were used for long-term machine learning. Neural mechanisms were simulated and a slightly complex representation schema was established. Because of the significant amounts of evolution that it has undergone, the natural visual system is highly efficient and optimized. The present practice of simulating a column map of V1 can offer a somewhat general representation and simplify downstream processing. Another difference from previous V1-inspired studies is that the present model was designed as a primary representation platform instead of a task-specific tool, and was upgraded to the level of semantic representation and data-concept dual driven processing.

There are always two important issues in computer vision and pattern recognition: 1) concept-driven processing and 2) the use of high-level knowledge. The root of semantic gap problem is breaking down the continuity of representations from raw data to formal concepts. The present conventional approach involves formalizing only some of these concepts according to the particular request of a task. In human cognition, however, there is no such discontinuity. Here, the array of simulated orientation columns in the primary visual cortex is sparser than that of raw data and denser than formal concepts. The semantic responsibilities assigned to this array are heavier than raw data and lighter than formal concepts. In this way, it is a very good place, with the right granularity, to allow for full data and concept converge. Raw data from downstream have been transformed into short lines (evidence), and concepts from upstream have been decomposed into short lines (hypothesis). They are all displayed in the same manner now, which greatly facilitates manipulation processes, such as selecting and organizing evidence, comparing hypotheses and evidence, and suggesting new hypotheses. This makes active processing and the translation of high-level concepts into low-level features possible. In the present model, learning an object, representing it, and extracting this information are considered under a common architecture. In this way, a possible separation between the definition of a piece of knowledge and the use of that knowledge can be avoided or mitigated.

Fig. 21 summarizes this paper and shows the whole course of transforming unstructured data into declarable knowledge. Future work should involve defining scale-extended geometrical attributes and introducing probability reasoning to provide more benefit in invariant recognition.

REFERENCES


[64] H. Wei and Y. Wang, “A new model to simulate the formation of orientation columns map in visual cortex,” in *Advances in Neural Networks (LNCS)*. New York, NY, USA: Springer-Verlag, 2011, pp. 35–41.


**Hui Wei** received the Ph.D. degree from the Department of Computer Science, Beijing University of Aeronautics and Astronautics, Beijing, China, in 1998. He was a Post-Doctoral Fellow with the Department of Computer Science and the Institute of Artificial Intelligence, Zhejiang University, Zhejiang, China, from 1998 to 2000. Since 2000, he has been with the Department of Computer Science and Engineering, Fudan University, Shanghai, China. His current research interests include artificial intelligence and cognitive science.

**Qiang Li** received the B.S. degree in computer science from Fudan University, Shanghai, China, in 2009, where he is currently pursuing the Ph.D. degree with the Department of Computer Science. His current research interests include cognition, brain-inspired computer vision, and neural network.

**Zheng Dong** received the B.S. and M.S. degrees in computer science from Fudan University, Shanghai, China, in 2008 and 2013, respectively. He is currently pursuing the Ph.D. degree with the Department of Computer Science. His current research interests include artificial intelligence and brain-inspired image processing.