

# АНАЛИЗ ИЗОБРАЖЕНИЯ, ПРИМЕНЯЕМЫЙ В МОДЕЛИ, ОБУЧАЮЩЕЙСЯ УЗНАВАНИЮ ЗРИТЕЛЬНЫХ ОБРАЗОВ

*А. П. Петров, Г. М. Зенкин*

## 1. Введение

В настоящее время авторами построена модель зрительного анализатора, обучающаяся узнаванию многих классов изображений [1]. Модель, выполненная в виде программы для ЭВМ, состоит из двух основных блоков: блока первичного анализа изображения и блока синтеза гештальт-операторов, в котором осуществляется процесс обучения модели распознаванию каждого из рассматриваемых классов. В предлагаемой работе описывается блок первичного анализа изображения, названный локальным анализом.

При построении модели имелось в виду, с одной стороны, использовать элементы, похожие на известные из электрофизиологических исследований зрительного анализатора, и с другой, — промоделировать некоторые особенности функционирования зрительной системы. Особое внимание при этом было обращено на способность людей и животных обнаруживать и узнавать объекты на сложном фоне.

К настоящему моменту проведены предварительные эксперименты с моделью. Модель обучена различению 25 классов изображений, включающих в себя буквы русского алфавита от А до Ф разнообразного написания и пять типов геометрических фигур (круги, эллипсы, треугольники, четырехугольники, невыпуклые многоугольники), а также обнаружению их на сложном фоне. Примеры изображений приведены на рис. 1, А, Б, В).

Для функциональной схемы модели, реализованной на ЭВМ, удобно пользоваться терминами, употребляемыми для описания нейронных сетей. При этом каждому элементу схемы будет соответствовать иногда достаточно большая подпрограмма модели.

## 2. Описание системы локального анализа

Входной растр, на который проектируется изображение, состоит из 1296 элементов ( $36 \times 36$ ), способных принимать два состояния — возбужденное (1) и невозбужденное (0). Края растра не сшивались, т. е. он был плоским, но дальнейшая обработка строилась таким образом, чтобы не

возникали краевые эффекты. Рис. 2 показывает, что на таком растре можно получать достаточно разнообразные изображения, которые и употреблялись в экспериментах с моделью.

**2. 1. Выделение контуров.** На начальном этапе анализа происходила предварительная обработка изображения, заключающаяся в том, что в



Рис. 1

модели выделялись два типа контуров, названных внешним и внутренним. Смысл этих операций виден на рис. 3.

Выделение контуров так же, как и следующий за ним локальный анализ изображения, выполнялись при помощи сумматоров с порогом, имеющих один выход типа «да — нет» и несколько входов. Такие сумматоры называются далее операторами. Каждый оператор вполне определялся рисунком соединения его входов с полем входного раstra и их весами, которые могли быть как положительными, так и отрицательными.

Внешний контур выделялся с помощью оператора, имеющего поле входов размером  $3 \times 3$  элемента, т. е. 9 входов, с центральным возбуждающим входом (вес его равен +8) и периферическими тормозящими входами (вес равен -1, рис. 4,а). Порог срабатывания оператора равен 1.

Легко видеть, что такой оператор срабатывает в тех местах раstra, где возбуждается центральный вход и не возбуждается хоть один из периферических входов. Такие условия всегда выполнены на границе черного и белого, т. е. в точках внешнего контура изображения (в частности, на линиях толщиной в два элемента).

Выделение внутреннего контура изображения — более сложная операция. Это связано, в частности, с тем, что само понятие внутреннего кон-

тура является довольно неопределенным. Однако это понятие можно сделать более точным, если ограничить его применение достаточно узким классом изображений. Среди них, например, будут изображения, состоя-

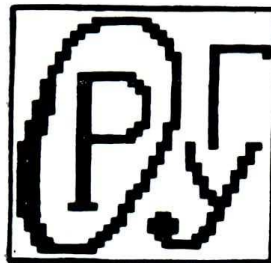
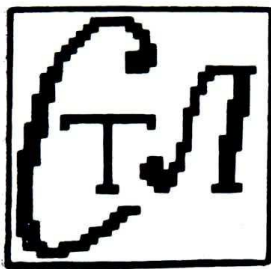


Рис. 2

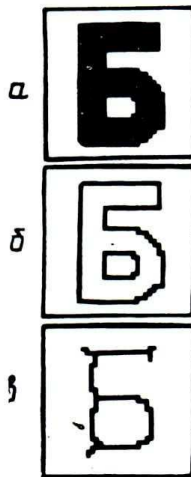


Рис. 3

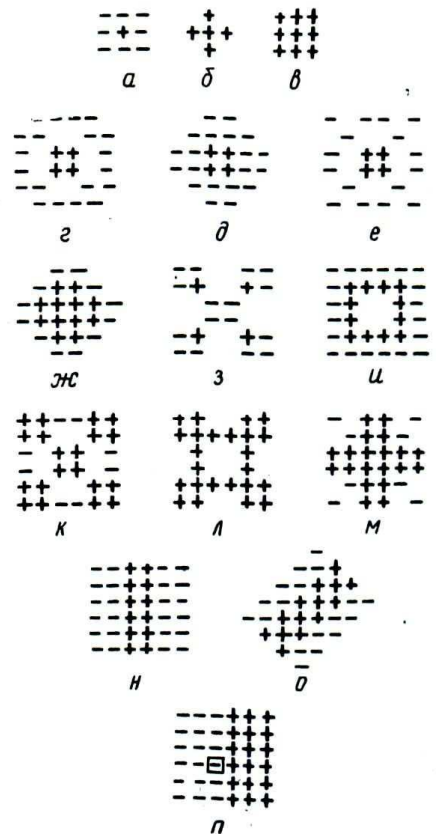


Рис. 4

щие из линий, длина которых значительно превосходит их толщину. Операция выделения внутреннего контура производится последовательным применением нескольких операторов, построенных по типу оператора внешнего контура.

Применяются следующие основные операторы.

1. Оператор сужения *С* имеет 9 входов, расположенных в виде квадрата  $3 \times 3$ . На рис. 4,б показано распределение весов его входов. Порог срабатывания оператора равен пяти. Оператор *С* возбуждается во всех точках изображения, кроме лежащих на границе черного и белого. Существенно отметить, что он не срабатывает на линиях толщиной в 1 и 2 элемента.

2. Оператор расширения *Р* имеет также поле входов  $3 \times 3$  элемента. Вес каждого его входа равен +1 (см. рис. 4,в), порог возбуждения оператора 1. Оператор расширения срабатывает в тех точках раstra, в ближайших окрестностях которых есть возбужденные элементы.

3. Оператор *Д* заполнения промежутков имеет поле входов  $3 \times 3$  и срабатывает в тех точках раstra, в ближайших окрестностях которых есть по крайней мере две возбужденных точки, не граничащие друг с другом.

Имеются также операторы с полем входов  $2 \times 2$ , подобные операторам *С*, *Р*, *Д*. Применяя все перечисленные операторы в каждой точке раstra, получаем поля их ответов. Каждый оператор может работать на поле от-

ветов другого, что можно условно записать, например, так:  $P(C)$  — оператор  $P$  срабатывает на поле оператора  $C$ .

Операция выделения внутреннего контура изображения тогда записывается следующим образом:

$$x_i = D((\sigma_{i-1} \dot{-} P(C(\sigma_{i-1}))) V C(\sigma_{i-1})) V (\sigma_{i-1} \dot{-} P(C(\sigma_{i-1}))) V x_{i-1}$$

где  $\sigma_0$  — растр с изображением;

$x_0$  — пустое поле;

$\sigma_i = C(\sigma_{i-1})$ ;  $i = 1, 2, 3, \dots, n$ ;

$V, \dot{-}$  обозначают операции логического сложения и сложения по модулю 2, выполняемые поточечно.

Число циклов  $n$  определяется условием:  $\sigma_{n-1}$  — не пустое, а  $\sigma_n$  — пустое поле. Поле  $x_n$  есть поле внутреннего контура.

После обработки входного растра описанными операторами на полях внешнего и внутреннего контуров еще могут присутствовать линии толщиной в два элемента растра. Дальнейшие операции, выполняемые операторами с полями входов  $2 \times 2$  элемента, превращали поля внешнего и внутреннего контуров в рисунок, линии которого имели толщину один элемент растра.

**2.2. Операторы локального анализа изображения.** С полем контура соединялись несколько типов операторов, осуществлявших локальный анализ изображения, помещенного на растр.

Операторы локального анализа так же, как и операторы выделения контуров, представляли собой сумматоры с порогом. Однако они имели по 36 ( $6 \times 6$ ) входов. Операторы разных типов отличались рисунками связей с элементами предыдущих полей (см. рис. 4,  $z-m$ ), весами входов (возбуждающий  $+1$ , тормозящий  $-1$ , нейтральный  $0$ ) и порогом. Применение оператора в каждой точке растра давало выходное поле оператора размером  $36 \times 36$ , на котором соблюдалось поточечное соответствие полю растра. Каждая точка выходного поля оператора, таким образом, определяла локальные свойства изображения в соответствующей точке растра.

Все операторы анализа изображения можно разбить на четыре группы: А) выделяющие локальные особенности на контуре изображения; Б) анализирующие ориентацию линий контура, В) кривых линий и Г) локальные особенности изображения по «ориентации».

Опишем каждую группу в отдельности.

**А. Операторы локальных особенностей** выделяли особенности трех типов: концы, углы и узлы. Поля входов этих операторов, а также пороги их срабатывания были получены в процессе предварительного отбора.

Выбор операторов производился только среди тех, поля входов у которых имели четыре оси симметрии. Ясно, что такой оператор полностью определяется заданием весов шести входов и порога срабатывания. Каждый вход мог иметь вес  $+1$ ,  $-1$  и  $0$ . Таким образом, все поля входов можно перенумеровать с помощью шестизначных троичных чисел. Всего их будет 729. Программа последовательно генерировала эти числа и строила из них поля входов операторов, а затем оценивала работу операторов с такими полями при различных порогах. С учетом различных порогов (от 0 до 8) получается всего 6560 операторов. Исключая операторы, состоящие из одних тормозящих входов, было проверено 5985 различных операторов.

Оценку операторов осуществляла программа, которая определяла качество их работы на группе контрольных изображений. Для каждого контрольного изображения было нарисовано по три маски, т. е. такого же размера картинка, где указывались места расположения особенностей со-

ответствующего типа (концов, углов или узлов). Требовалось, чтобы оператор соответствующего типа возбуждался главным образом в местах, указанных на маске.

Работа оператора оценивалась сравнением выходного поля оператора с полями масок. Этим определялось качество оператора, которое считалось наивысшим, если оператор возбуждался только в местах, указанных на одной маске. Тип этой маски определял и тип оператора. С достаточно высоким качеством программа отобрала примерно по 40 операторов каждого типа. Дальнейший выбор был произведен более подробным анализом их работы самими экспериментаторами.

По некоторым причинам, в частности из-за сравнительно простого устройства поля входов оператора и неполной его симметрии, один оператор не мог выделять все локальные особенности своего типа. Однако несколько операторов, отобранных для выделения локальных свойств одного типа, справлялись с этой задачей вполне удовлетворительно. Было исследовано несколько способов объединения операторов одного типа, и самым простым и надежным оказалось логическое сложение их выходных полей. В модели для выделения локальных особенностей различных типов употреблялось по 3—5 операторов. Примеры устройства полей входов отобранных операторов приведены на рис. 4, *г—м*.

**Б. Операторы ориентированных линий** выделяли отрезки линий четырех ориентаций. Устройство полей входов операторов, выделяющих вертикальные и наклонные отрезки линий, показано на рис. 4, *н—о*. Два других оператора получаются из этих поворотом на  $90^\circ$ . Операторы, похожие на эти, были обнаружены в зрительной коре кошки [2].

Порог возбуждения операторов был подобран в эксперименте и был достаточно высок для того, чтобы они не возбуждались на участках линий с большой кривизной.

**В. Анализ участков линий с большой кривизной** производился операторами ориентированных кривых. Эти операторы по своему устройству совпадают с операторами ориентированных линий, но отличаются от них более низким порогом. В отличие от предыдущих операторов анализа операторы ориентированных кривых применяются не к полю контура, а к полю кривых линий. Оно получается после удаления из поля контура всех выделенных ранее частей (концов, углов, узлов и ориентированных линий).

**Г. Операторы ориентированных локальных особенностей** проводят классификацию концов, углов и узлов в зависимости от их ориентации на растре. Ориентация того или иного участка контура, на котором возбуждился один из операторов концов, углов или узлов, определяется взаимным расположением этой особенности и отрезков линий, ее образующих.

При этом так же, как и в случае ориентированных линий, различается только четыре направления: вверх — вниз, вправо — влево. Поля входов одного из этих операторов изображены на рис. 4, *п*. Такое устройство операторов приводит к тому, что например, конец наклонной линии будет иметь две ориентации, т. е. на нем сработают два типа операторов ориентированных локальных особенностей. Может оказаться, что некоторая особенность не имеет ориентации: например, так может произойти с центрально симметричным узлом (крест).

Операторы ориентированных локальных особенностей работают на поле контура, но они могут возбуждаться только при попадании на соответствующую особенность изображения, т. е. каждый оператор имеет еще один вход, который соединен с полем выходов операторов концов, углов и узлов.

На этом локальный анализ изображения оканчивается. Всего получается 24 поля анализа, не считая самого изображения и его контура. Эти поля следующие:

3 поля локальных особенностей (концов, углов и узлов);

4 поля ориентированных линий;

1 поле «кривых» линий;

4 поля ориентированных кривых линий;

12 полей (3×4) ориентированных локальных особенностей.

Пример проведенного в эксперименте анализа специально выбранного изображения приведен на рис. 5.

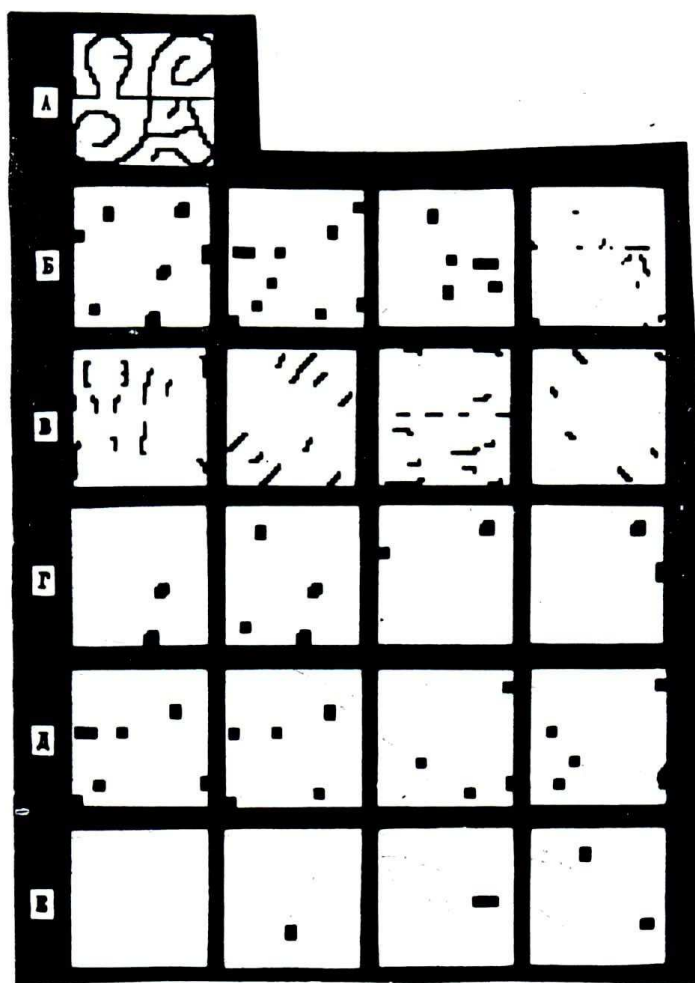


Рис. 5

### 3. Использование локального анализа

Легко видеть, что первичный анализ представляет собой классификацию всех точек изображения, учитывающую их взаимное расположение в пределах очень небольшого поля входов оператора (не более  $7 \times 7$  элементов). Именно поэтому такой анализ является локальным.

Существенно отметить, что локальный анализ сохраняет информацию о взаимном расположении классифицированных описанным выше образом точек изображения. Для решения интересующего нас класса задач необходимо было использовать эту информацию.

Были проверены два пути использования этой информации: описание полей анализа набором числовых параметров и построение дальнейшего анализа взаимного расположения частей с помощью нелокальных пространственных операторов, построенных по типу описанных выше сумматоров с порогом. Эксперименты показали, что для решения тех задач, которые были предложены системе, достаточно было сформировать лишь один нелокальный оператор на каждый класс изображений, примеры которых приведены на рис. 1, а. Формирование таких операторов происходило в блоке синтеза. Эти операторы были названы гештальт-операторами. Таким образом, обучение модели заключалось в синтезе гештальт-оператора, представляющего собой память («эталон») одного класса. Подробнее эта часть модели описана ранее [1].

Здесь мы опишем первый путь использования локального анализа. Были проведены специальные эксперименты, в которых описание всех 24 полей анализа производилось при помощи некоторых числовых параметров, в качестве которых были взяты: число фигур на поле, координаты центра «тяжести» черных точек, общий размер изображения на поле и число черных элементов («площадь» изображения). Общее число параметров для характеристики каждого входного изображения было равно 89.

Все эти характеристики каждого поля анализа поступали далее в блок выбора решающей функции, которая должна принимать значения 1, 2, ...,  $N$  ( $N$  — число классов). Блок выбора решающей функции состоял из двух частей. Во-первых, из всех 89 параметров выбирались так называемые минимальные допустимые совокупности, содержащие не более 5—6 параметров. Затем в результате их анализа строились функции, которые употреблялись как признаки того или иного класса. При узнавании все отобранные признаки суммировались, и ответ определялся по наибольшему числу «голосов».

Алгоритм выбора минимальных допустимых совокупностей переменных, предложенный авторами для решения задач узнавания образов в 1963 г., можно описать следующим образом. В результате локального анализа и получения числовых параметров полей, описание входного изображения можно представить в виде кортежа  $\langle c_1, c_2, c_3, \dots, c_L \rangle$  ( $L$  — число параметров). Для дальнейшего нам будет удобно сделать одно предположение, которое упростит описание алгоритма почти не изменив его смысл. Пусть мы имеем систему свойств (параметров)  $\langle c_1(x), c_2(x), \dots, c_L(x) \rangle$ . Опираясь на них, мы можем определить систему задач  $\{M_\alpha\}$  следующим образом. Возьмем любой набор свойств, например,  $c_{i_1}, c_{i_2}, \dots, c_{i_l}$ , и построим разбиение множества объектов  $x \in X$ , отнеся к одному классу объекты, неразличимые в этом наборе свойств, т. е.  $x_1 \equiv x_2$ , если

$$c_{i_1}(x_1) = c_{i_1}(x_2)$$

$$c_{i_2}(x_1) = c_{i_2}(x_2)$$

.....

$$c_{i_l}(x_1) = c_{i_l}(x_2).$$

Это и есть одна задача. Рассмотрев все такие наборы из выбранной совокупности свойств, получим систему задач  $\{M_\alpha\}$ . Таким образом, в этой системе задач будут, например, разбиения, порождаемые отдельными свойствами  $c_i(x)$ , парами свойств  $\langle C_1, C_2 \rangle$  и т. д. Предположе-

ние будет состоять в том, что все задачи, которые необходимо нам решать, содержатся среди задач системы  $\{M_\alpha\}$ \*

Пусть теперь системе представлены объекты для обучения, расклассифицированные согласно выбранной задаче  $M_\alpha$ . Чтобы на основании приведенных в обучении объектов указать набор соответствующих свойств, образуем для каждой пары объектов (из обучения), принадлежащих разным классам, так называемый код сравнения:

$$\langle r_1, r_2, \dots, r_L \rangle, \quad (1)$$

где

$$r_i = \begin{cases} 0, & \text{если } C_i(x_1) = C_i(x_2) \\ 1, & \text{если } C_i(x_1) \neq C_i(x_2) \end{cases} \quad i=1, 2, \dots, L.$$

Рассмотрим теперь совокупность всех образованных кодов сравнения. Очевидно, что если выбранное разбиение совпадает с разбиением по набору  $C_{i_1}, C_{i_2}, \dots, C_{i_L}$  свойств, то в любом из наших кодов сравнения хотя бы одно из чисел  $r_{i_1}, r_{i_2}, \dots, r_{i_L}$  отлично от 0, т. е.

$$r_{i_1} + r_{i_2} + \dots + r_{i_L} > 0. \quad (2)$$

Любую совокупность свойств, для которых выполняется выражение (2) в каждом коде сравнения, мы назовем допустимой, имея в виду возможность того, что выбранной задачей является  $M_\alpha$ , соответствующая этой совокупности свойств.

Если к допустимой совокупности свойств добавить новое свойство, то получившаяся совокупность останется допустимой. Если же из допустимого набора исключить некоторое свойство, то, вообще говоря, оставшаяся совокупность может стать недопустимой. Мы будем искать все минимальные допустимые наборы свойств, т. е. наборы, из которых уже нельзя исключить ни одного свойства.

Алгоритм поиска минимальных допустимых совокупностей свойств будет основан на следующих двух очевидных фактах:

1° для любого кода сравнения всякая минимальная допустимая совокупность свойств содержит хотя бы одно из свойств, отмеченных в этом коде сравнения единицей;

2° рассмотрим набор свойств  $C_{i_1}, C_{i_2}, \dots, C_{i_L}$ , целиком содержащийся в какой-нибудь минимальной допустимой совокупности; пусть  $\langle r_1, r_2, \dots, r_L \rangle$  — код сравнения, в котором  $r_{i_1} + r_{i_2} + \dots + r_{i_L} = 0$ . Тогда среди остальных свойств минимальной допустимой совокупности есть по крайней мере одно свойство, отмеченное в этом коде  $\langle r_1, r_2, \dots, r_L \rangle$  единицей.

Конкретная организация алгоритма может быть очень различной. Мы покажем один из возможных вариантов на примере. Девять строк таблиц являются кодами сравнения ( $L=5$ ), расположенными в порядке возрастания в них числа единиц:

|   |   |   |   |   |
|---|---|---|---|---|
| 0 | 1 | 0 | 0 | 1 |
| 0 | 0 | 1 | 0 | 1 |
| 1 | 0 | 1 | 0 | 0 |
| 0 | 1 | 0 | 1 | 0 |
| 0 | 0 | 1 | 1 | 1 |
| 1 | 1 | 0 | 0 | 1 |
| 0 | 1 | 0 | 1 | 1 |
| 1 | 0 | 1 | 0 | 1 |
| 1 | 1 | 0 | 1 | 0 |

\* Можно дополнить алгоритм, приведенный ниже так, чтобы снять это ограничивающее предположение.



Берем первый код сравнения 01001. Выписываем по отдельности каждое отмеченное в нем свойство. Это можно представить так:

0 0 0 0 1  
0 1 0 0 0

Добавляя к каждому из этих свойств другие, мы построим все минимальные допустимые совокупности свойств (см. пункт 1°). В силу пункта 2° мы можем искать добавочные свойства для первого выписанного свойства 00001 среди свойств кода 10100\* (третий код сравнения в таблице), а для второго 01000 — среди свойств  $C_3$  и  $C_5$ , отмеченных единицей во втором коде сравнения 00101. Выпишем теперь по отдельности все получающиеся таким образом пары свойств:

0 0 1 0 1  
1 0 0 0 1  
0 1 1 0 0  
0 1 0 0 1

Для каждой из выписанных четырех пар свойств ищем в таблице код сравнения, в котором соответствующие свойства не отмечены единицами. При этом оказывается, что для третьей пары свойств 01100 такого кода нет. Это означает, что свойства  $C_2$  и  $C_3$  образуют одну из искомым минимальных допустимых совокупностей свойств. Из остальных пар и найденных для них кодов сравнения получаем совокупности, состоящие из трех свойств: 00111, 10011, 11001. Все они уже являются минимальными допустимыми совокупностями свойств.

Опыты с использованием этого алгоритма проводились на буквах А, Б, В, Г (см. рис. 1, а). Было проведено несколько вариантов обучения и контрольного узнавания. Эксперименты показали, что такой способ использования анализа не является удовлетворительным (программа давала 25—30% ошибочных ответов, а при случайном угадывании должно быть 75%). Но самым интересным явлением оказалась зависимость числа переменных в решающей функции от количества букв, данных в обучении. Это число росло вместе с объемом обучения.

Анализ показал, что при использовании в обучении вместо наборов параметров объектов наборов случайных значений параметров число переменных в отобранных функциях не увеличивается. Отсюда можно сделать вывод, что если и существует решающая функция от выбранных параметров, то она зависит от очень большого числа переменных.

Таким образом, попытки воспользоваться набором каких-либо числовых параметров, описывающих каждое поле анализа изображения, по-видимому, не могут привести к успеху, что и подтвердили проведенные с моделью эксперименты. Это происходит прежде всего из-за сильной зависимости между параметрами (свойствами). Пользуясь терминологией статьи В. С. Файна [3], можно сказать, что точки каждого класса в пространстве параметров заполняют некоторые причудливо расположенные нити, а не объемы. Характеристические функции таких нитей необходимо содержат весьма большое число переменных (параметров). А это приводит к необходимости проделать при обучении огромный перебор и проверку возможных решающих функций.

#### ЛИТЕРАТУРА

1. Зенкин Г. М., Петров А. П. Система анализа изображения и узнавание объектов на сложном фоне. Биофизика, 1967, № 3.
2. Hubel D. H., Wiesel T. N. 1959. Receptive fields of single neurones in the cat's striate cortex. Journ. Physiol., 148, 574.
3. Файн В. С. К определению понятия «сходство». Изв. АН СССР, Техническая кибернетика, 1966, № 2.

\* Т. е. свойств  $C_1$ ,  $C_3$ , отмеченных единицами в коде 10100.