УО «ВГУ им. П.М. Машерова»
Кафедра прикладной математики и механики

Е.А. Корчевская

Распознавание образов

Методические материалы по дисциплине «Распознавание образов» для студентов 4 курса специальности Прикладная математика (1-31 03 03)

Витебск, 2011

Лабораторная работа №1.

Задание: распознать букву, которая подается на вход в виде изображения.

Методы решения: Метод эталонов, Метод дробящихся эталонов.

Решение оформить в виде программы на любом языке программирования, на вход которой подается черно-белое изображение буквы любого размера.

Также необходимо представить отчет.

Лабораторная работа №2.

Задание. Структурным методом распознать геометрические фигуры (ромб, круг, квадрат, треугольник).

Указания к работе.

Чтоб программе "понять" что за фигура представлена, нужно описать параметры этой фигуры. Допустим программа должна распознавать треугольник, квадрат, круг, ромб. Особенностью их параметров должны быть наличие искривлений: острых, гладких. Для этого начинаем анализировать поверхность границы фигуры, ищем различные углы и искривления. Соответственно для треугольника мы найдем 3 явных острых искривлений, для квадрата и ромба 4, для круга 0, но выявится искривление под постоянным углом (производная от функции задающей границу в полярной СК приближенно является константой"). Такого искривления не будет наблюдаться у других выше перечисленных фигур, иначе они будут интерпретироваться программой как другие фигуры. Далее проверяем другие имеющиеся параметры, определяем длины отрезков(кривых), величины углов(зная координаты вершин это сделать очень просто), ну а потом сверяем с базой знаний, что это будет за объект.

Лабораторная работа №3.

Задание.

Реализовать преобразование Хафа для поиска прямых, окружностей и эллипсов на изображение.

Указание к лабораторной работе.

1. Линейное преобразование Радона

Преобразование Радона R (k,b) непрерывной функции f (x,y) вычисляется путём интегрирования (сложения) значений f вдоль наклонной линии, как показано на рисунке 1:

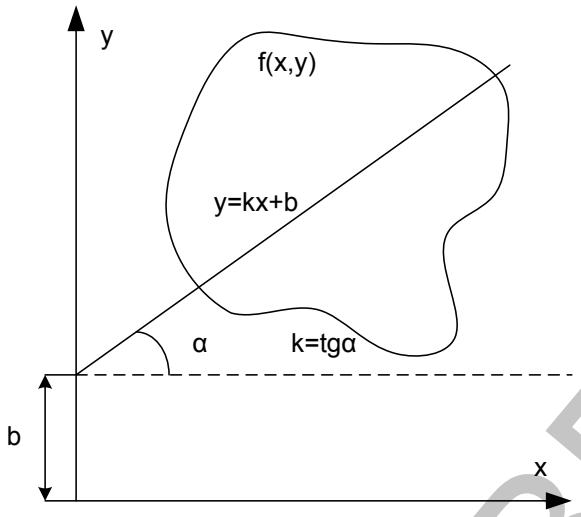


Рисунок 1. Линейное преобразование Радона

Можно записать:

$$R(k,b) = \int_{-\infty}^{\infty} f(x, kx+b)dx \tag{1}$$

Или при помощи δ-функции Дирака:

$$R(k,b) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(x,y)\delta(y-kx-b)dxdy \tag{2}$$

Необходимо отметить, что преобразование (1) или (k,b)-преобразование обладает некоторыми свойствами, очень важными для работы с изображениями, такими как свойство линейности (3), сдвига (4), масштабирования (5).

Свойство линейности можно сформулировать следующим образом: «Преобразование Радона взвешенной суммы функций равно взвешенной сумме преобразований каждой функции»:

$$R\{\sum w_i f_i(x,y)\} = \sum w_i R\{f_i(x,y)\} \tag{3}$$

Свойства (4) и (5) (сдвиг и масштабирование) показывают, как вычисляется (k,b)-преобразование при изменении аргументов интегрируемой функции.

$$R\{f(x-\tilde{x}, y-\tilde{y})\} = R(k, b-\tilde{y}+k\tilde{x}) \tag{4}$$

$$R\{f(\frac{x}{n}, \frac{y}{m})\} = nR(\frac{kn}{m}, \frac{b}{m}) \tag{5}$$

Рассмотрим несколько элементарных примеров:

Любую точку можно представить в виде произведения 2-х δ-функций:

$$f(x,y) = \delta(x)\delta(y) \tag{6}$$

Тогда её преобразование Радона будет иметь вид:

$$R(k,b) = \int_{-\infty}^{\infty} \delta(x)\delta(kx+b)dx = \delta(b) \tag{7}$$

Пользуясь свойством сдвига, получим:

$$f(\tilde{x},\tilde{y}) = \delta(x-\tilde{x})\delta(y-\tilde{y}) \tag{8}$$

Преобразование Радона в этом случае:

$$R(k,b) = \delta(b-\tilde{y}+k\tilde{x}) \tag{9}$$

Таким образом, преобразование Радона точки имеет вид прямой (рисунок 2).

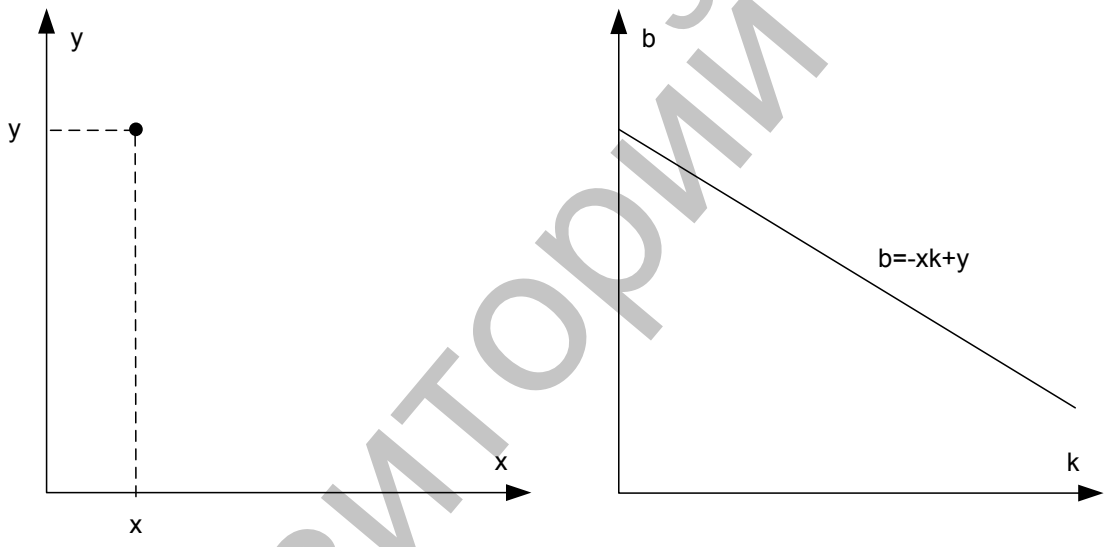


Рисунок 2. Преобразование отдельной точки.

Стоит отметить этот вывод, поскольку любая функция может быть представлена в виде взвешенной суммы (интеграла) множества точек.

Соответственно, для прямой линии, заданной уравнением $y=kx+b$ получим:

$$f(x,y) = \delta(y-\tilde{k}x-\tilde{b}) \tag{10}$$

Преобразование Радона в этом случае:

$$R(k,b) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \delta(y-\tilde{k}x-\tilde{b})\delta(y-kx-b)dxdy = \int_{-\infty}^{\infty} \delta((k-\tilde{k})x+b-\tilde{b})dx \tag{11}$$

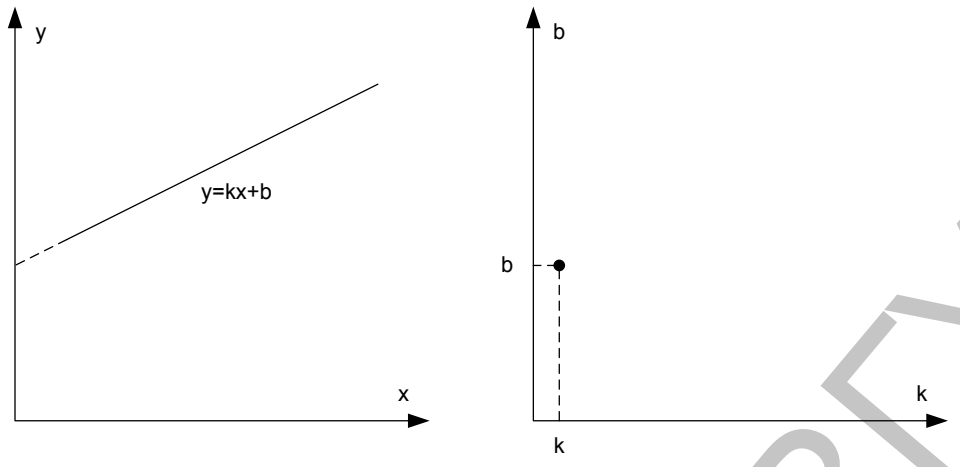


Рисунок 3 . Преобразование прямой линии.

2. Дискретное преобразование Радона.

Для вычисления на ЭВМ необходимо провести дискретизацию. Самый простой способ – линейная выборка значений x и y:

$$\begin{aligned} x &= x_m = x_{\min} + m\Delta x, m = \overline{0, M-1} \\ y &= y_n = y_{\min} + n\Delta y, n = \overline{0, N-1} \end{aligned} \tag{12}$$

Тогда преобразование Радона аппроксимируется простым суммированием:

$$\mathrm{R}(\mathrm{k}_t, \mathrm{b}_h) \approx \Delta x \sum_{m=0}^{M-1} f(x_m, k_t x + b_h) \tag{13}$$

Поскольку y – целое число, возникает проблема интерполяции значений y:

$$y_n \neq b_t x_m + t_h \tag{14}$$

Два самых простых подхода – интерполяция ближайшего соседства и линейная интерполяция:

Реализация алгоритма:

✓ Переменной x назначаются дискретные значения из рассматриваемого интервала. Затем вычисляется соответствующее значение переменной y.

✓ В случае интерполяции ближайшего соседства значение *y* округляется до ближайшего целого. Вычислительная сложность алгоритма в этом случае будет порядка *0(T,H,M)*. где *T, H* и *M* количество отсчётов *k, x* и *b* соответственно. Также будет иметь место погрешность округления (см. рисунок 4).

✓ При линейной интерполяции между двумя соседними отсчётами функции погрешность вычислений будет меньше при той же вычислительной сложности алгоритма (см. рисунок 5).

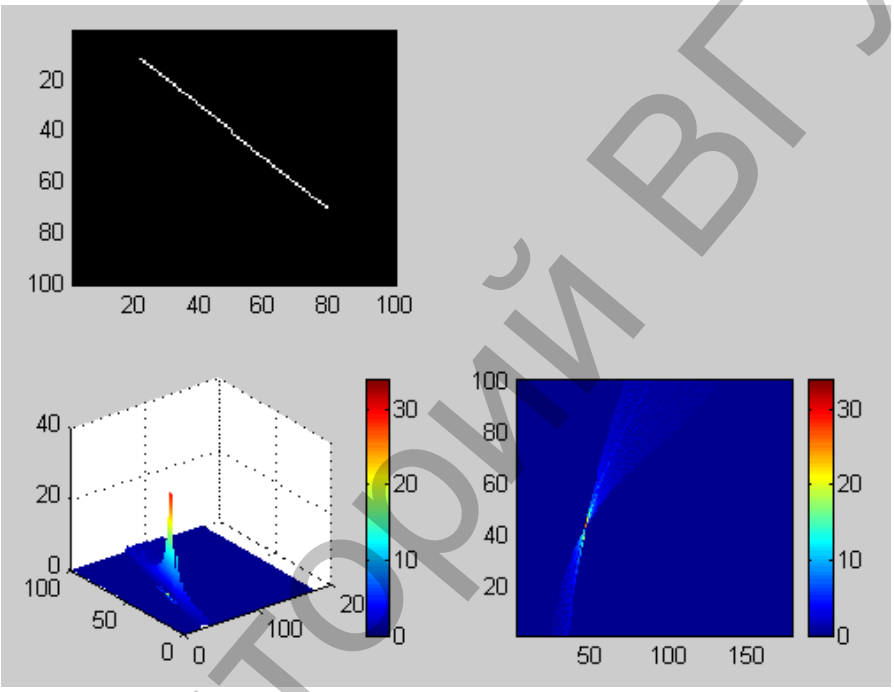


Рисунок 4. Вычисление преобразования Радона прямой с интерполяцией ближайшего соседства.

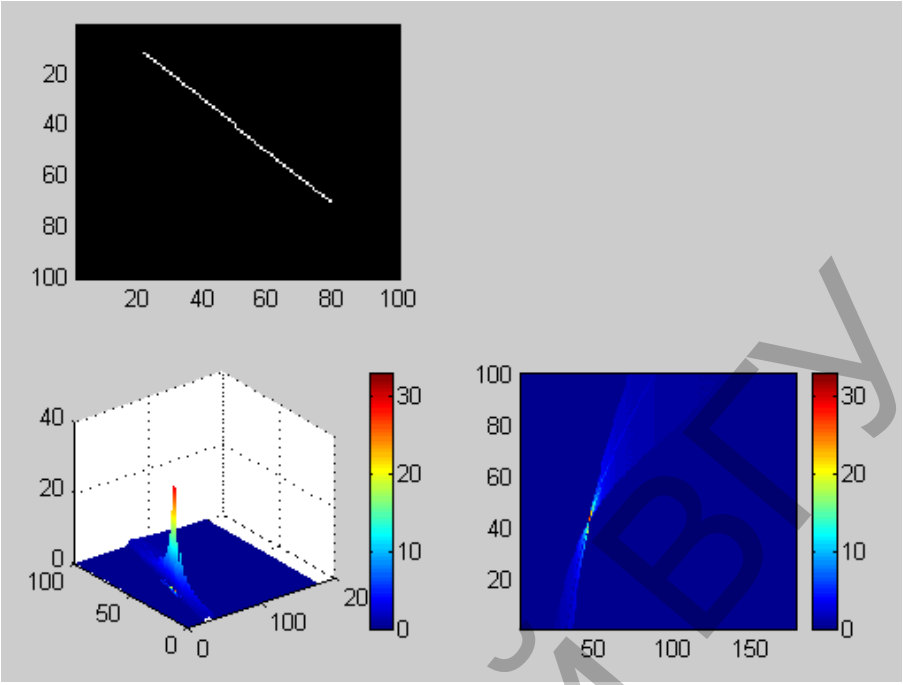


Рисунок 5. Вычисление преобразования Радона прямой с линейной интерполяцией.

Далее приводятся тексты соответствующих функций для MatLab

```
//Интерполяция ближайшего соседства
   P = tan(-pi*89/180:pi/180:pi*89/180);
   R = length(P);
   TAU = -N/2:N/2;
   T = length(TAU);
   result = zeros(T, R);

   for th = 1:T
      for p = 1:R
         for x = -T:T
            y = x*P(p) + TAU(th);
            yy = T-round(y);
            xx = T+x;
            if (xx>0) && (xx<=N) && (yy>0) && (yy<=N)
               result(th, p) = result(th, p) + IMG(xx, yy);
            end
         end
      end
   end

//Линейная интерполяция
 TAU = -N/2:N/2;
   T = length(TAU);
   P = tan(-pi*89/180:pi/180:pi*89/180);
```

```
R = length(P);

result = zeros(T, R);

for p = 1:R
   for th = 1:T
      for x = -T:T
         xx = T+x;
         y = x*P(p) + TAU(th);
         y1 = floor(y);
         w = y - y1;
         yy1 = T - y1;
         yy2 = T - (y1+1);
         if (xx>0) && (xx<=N) && (yy2>0) && (yy1<=N)
result(th, p) = result(th, p) + IMG(xx, yy1)*(1-w) + IMG(xx, yy2)*w;
         end
      end
   end
end
```

3. Нормальное преобразование Радона.

Нормальное уравнение прямой (рисунок 6) имеет вид:

$$\rho = x\cos\theta + y\sin\theta \qquad (15)$$

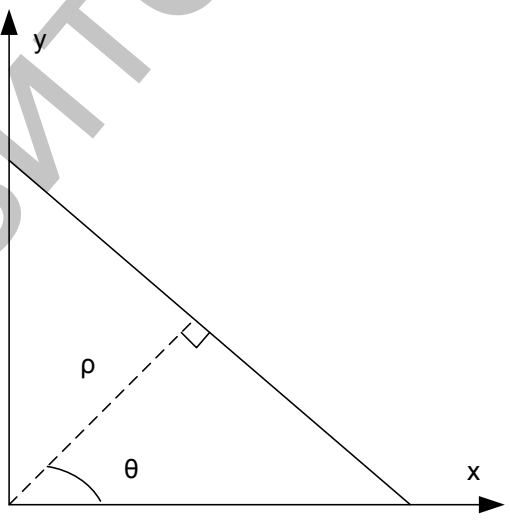


Рисунок 6. Нормальное уравнение прямой.

Аналогичным образом, преобразование Радона можно вычислить по формуле:

$$R(\rho,\theta) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(x,y)\delta(\rho - x\cos\theta - y\sin\theta)dxdy$$

(16)

Простейшие случаи – преобразование точки и прямой линии (17) и (18)

$$R(\rho,\theta)=\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\delta(x-\tilde{x})\delta(y-\tilde{y})\delta(\rho-x\cos\theta-y\sin\theta)dxdy=\delta(\rho-\tilde{x}\cos\theta-\tilde{y}\sin\theta) \quad (17)$$

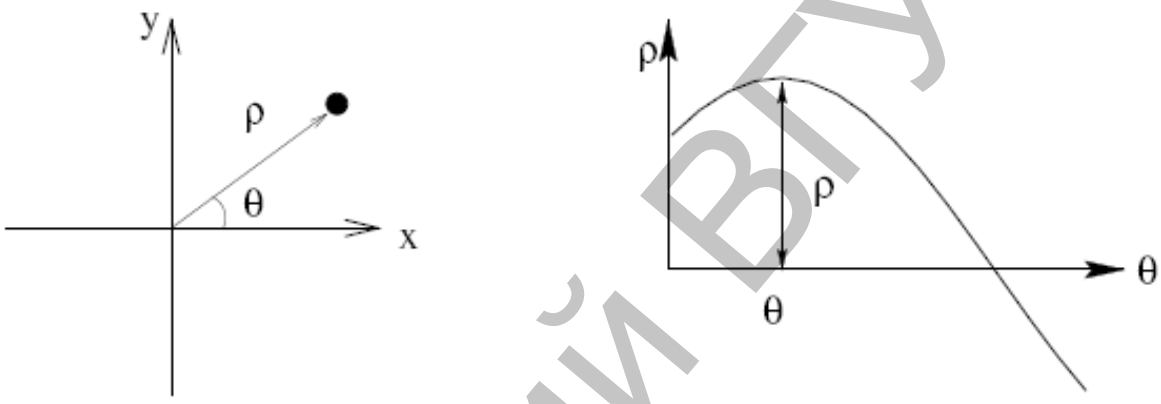


Рисунок 7. Нормальное преобразование точки

$$R(\rho,\theta)=\int_{-\infty}^{\infty}\delta(\tilde{\rho}-(\rho\cos\theta-s\sin\theta)\cos\tilde{\theta}-(\rho\cos\theta+s\sin\theta)\sin\tilde{\theta})ds=\int_{-\infty}^{\infty}\delta(\tilde{\rho}-\rho\cos(\theta-\tilde{\theta})+s\sin(\theta-\tilde{\theta}))ds \quad (18)$$

4. Связь с преобразованием Фурье.

Одномерное преобразование Фурье по переменной *s* от преобразования Радона функции *f* даёт двумерное преобразование Фурье функции *f*.

$$F(\omega\cos\alpha,\omega\sin\alpha)=\int_{-\infty}^{\infty}dse^{-i\omega s}R(s,\alpha) \quad (19)$$

Поскольку двумерное преобразование Фурье обратимо, то обратимо и преобразование Радона (20).

$$f(x,y)=\int_0^{2\pi}d\alpha\int_0^{\infty}\frac{\omega d\omega}{(2\pi)^2}e^{i\omega(x\cos\alpha+y\sin\alpha)}\tilde{R}(\omega,\alpha) \quad (20)$$

где:

$$\tilde{R}(\omega,\alpha)=\int dse^{-i\omega s}R(s,\alpha)$$

Выражение (20) есть формула обращения преобразования Радона.

К сожалению, преобразование Радона обратимо не для каждой функции.

5. Примеры

В качестве иллюстрации можно привести пример преобразования изображения, содержащего несколько прямых линий.

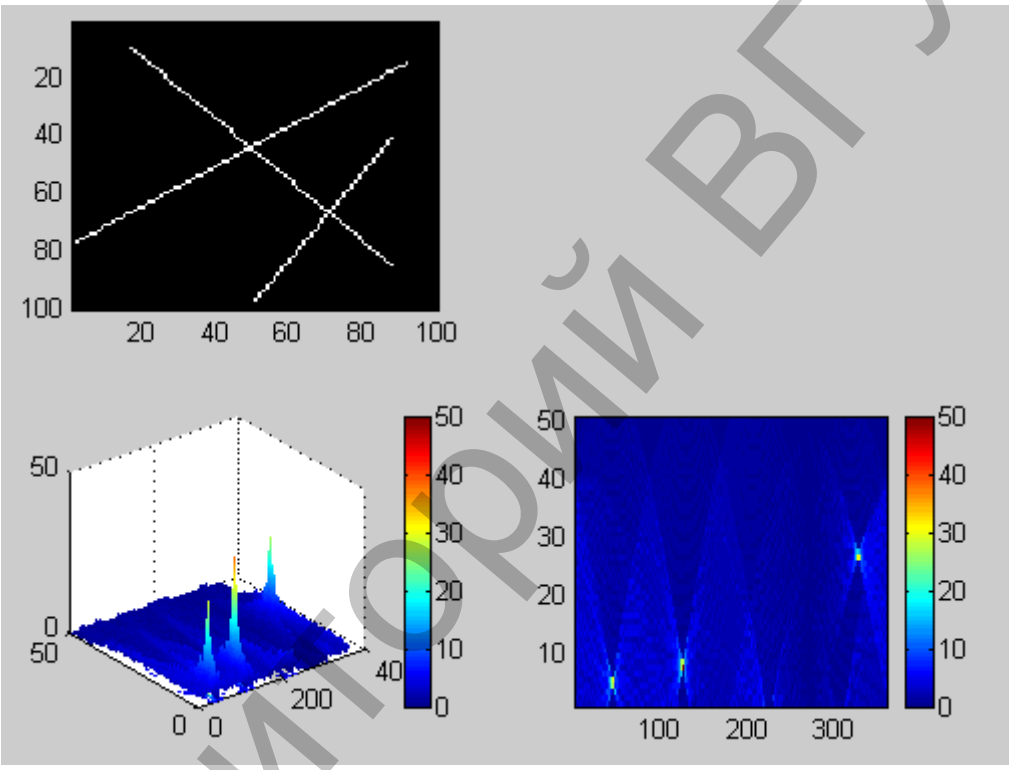


Рисунок 8. Нормальное преобразование нескольких прямых.

Как видно на рис.8, рассматриваемым прямым в результирующем пространстве признаков соответствуют пики значений интегрирования. При дальнейшей обработке полученного результирующего пространства получим точки, соответствующие прямым. Очевидна практическая ценность – данную модификацию преобразования можно использовать для обнаружения прямых (прямолинейных отрезков) на изображении.

Необходимо также отметить, что нормальное преобразование Радона подходит для выявления прямых любой ориентации, что невозможно при линейном преобразовании (прямые, параллельные оси У).

Рассмотрим зашумлённое изображение:

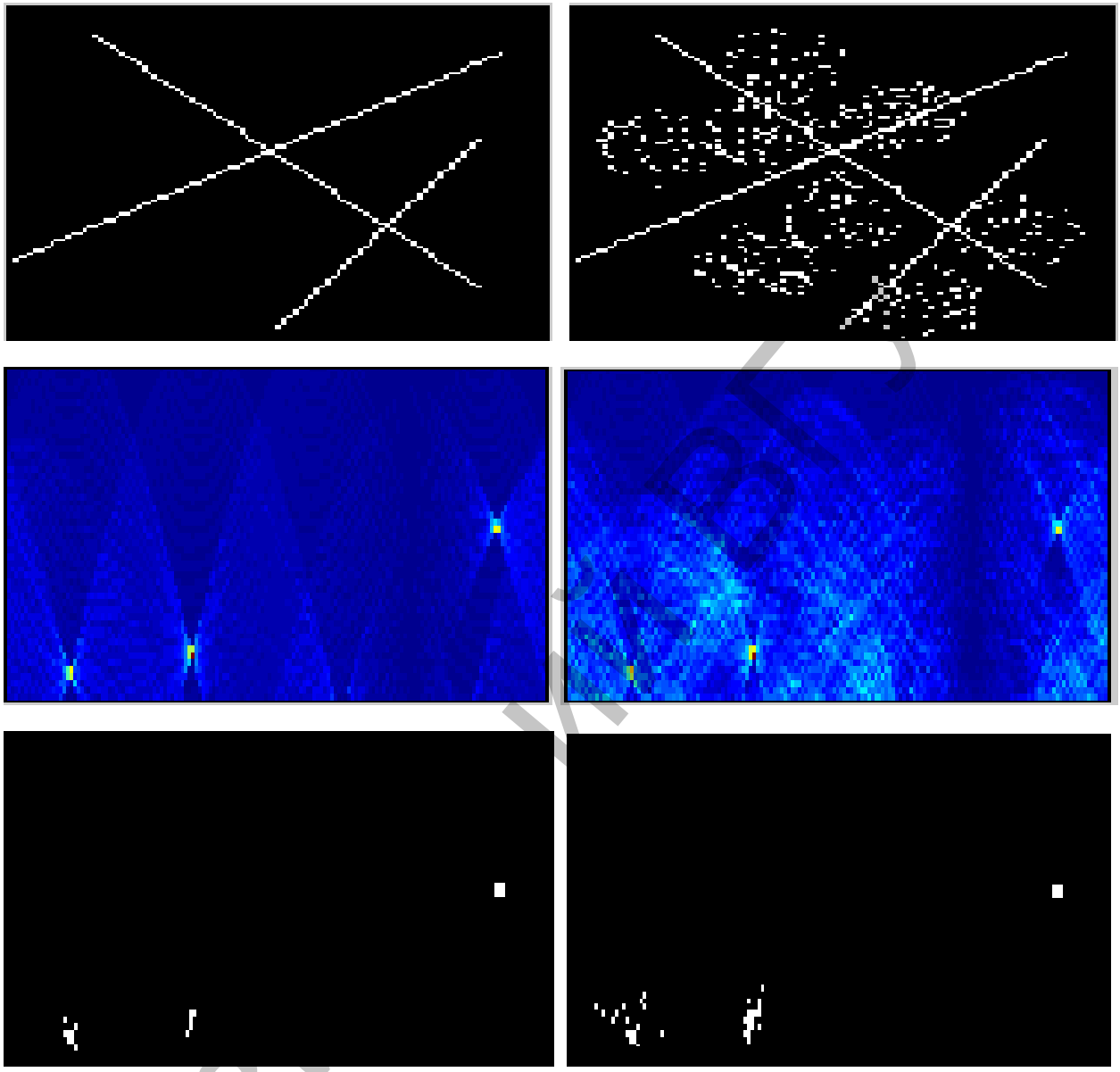


Рисунок 9. Работа с зашумлённым и незашумлённым изображением

Итак, как показано на рис. 9 (нижний ряд), преобразование Радона практически с одинаковой точностью позволяет определить расположение прямых на зашумлённых и незашумлённых изображениях. Это свойство преобразования в значительной степени расширяет область его применения. Например, можно применять преобразование для анализа зашумлённых полутоновых изображений интегральных микросхем.

6. Преобразование Хафа.

При анализе монохромного изображения при использовании преобразования Радона, как описано ранее, исходное изображение отображается в некоторое пространство признаков (так для (k,b) – преобразования это про-

странство параметров прямой k и b, а для нормального – ρ и θ соответственно). Затем по виду результирующего изображения можно судить о наличии на исходном изображении определённых объектов, их формы и т.д. Как отмечалось ранее, при реализации такого преобразования появляются определённые сложности, связанные с необходимостью “подгонять” значения аргументов под рассматриваемые параметры. Такой необходимости не будет, если анализировать не исходное изображение, а результирующее пространство. Каждой точке его можно поставить в соответствие счётчик, и затем, анализируя результаты аналогичного преобразования наращивать счётчики. Таким образом, анализируя счётчик каждой точки результирующего пространства, представляющей собой ни что иное, как представление фигуры интереса (искомой фигуры, кривой и т.д.), можно судить о наличие фигур с задаваемыми параметрами.

Такое преобразование называется преобразованием Хафа. Его идея состоит в выявлении кривых заданных определёнными параметрами:

$$F(a_1, a_2, \ldots, a_n, x, y) = 0 \qquad (21)$$

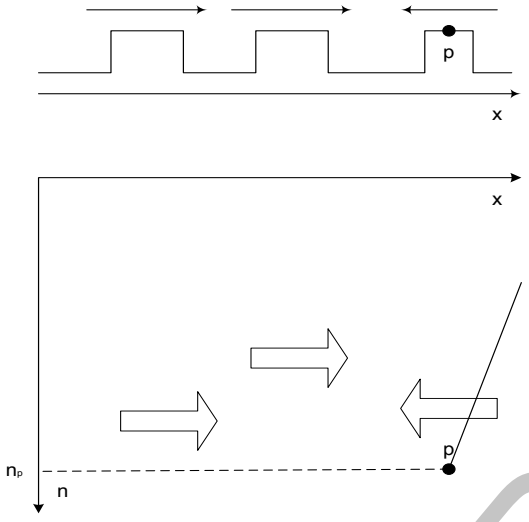
Пространство, содержащее результирующее изображение называют фазовым. Его образуют параметры семейства кривых $a_1, a_2, \ldots a_n$, задаваемого формулой (19).

Алгоритм поиска заданных прямых очень прост и состоит лишь в вычислении значений счётчиков фазового пространства, а затем, в поиске наибольших их значений.

Существует несколько модификаций вычисления преобразования Хафа.

- ✓ Случайное преобразование – заключается в анализе случайным образом выбранных точек.
- ✓ Иерархическое преобразование – исходное изображение делиться на небольшие квадраты, анализируется, затем квадраты группируются по 4 в большие квадраты и т.д.
- ✓ «Размытие» фазового пространства – в случае зашумлённых изображений, для увеличения точности наращиваются счётчики не только рассматриваемой точки фазового пространства, но и прилежащие ей счётчики близлежащих кривых.
- ✓ Использование градиента яркости – рассматриваются только кривые перпендикулярные градиенту яркости изображения в данной точке, если оно получено операторами выделения краёв.

Преобразование Хафа может применяться там, где необходимо анализировать формы кривых. Это всевозможные системы распознавания образов, таких как рукописный шрифт, анализ изображений микросхем, аэрокосмических снимков а также системы слежения (рис. 10).



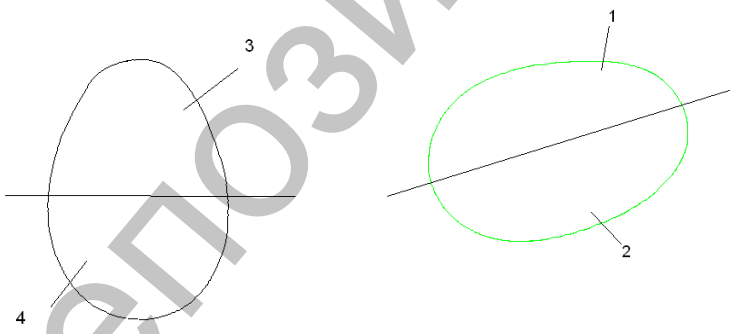
Пусть x – одномерная последовательность, полученная после предварительной обработки видео изображения. Значение n определяет номер кадра. Траектория движения точки p – некоторая кривая.

Если движение точки равномерно, то кривая есть прямая линия, задаваемая уравнением:

$$x = k(n - n_p) + b \qquad (22)$$

Лабораторная работа №4.

С помощью сплайна третьей степени произвести интерполяцию 1, 2, 3, 4 областей контуров.



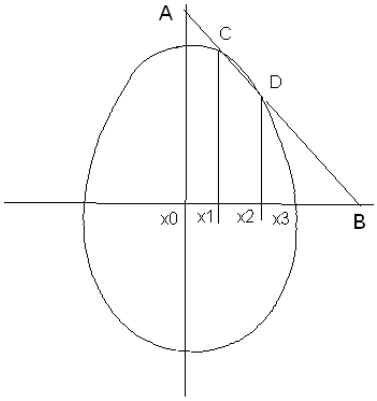
Для каждого контура из соответствующих папок вычислить коэффициенты интерполяционного сплайна третьей степени и занести данные в таблицу.

Лабораторная работа №5.

Формирование признакового пространства.

Даны контуры объектов. Правую верхнюю четверть необходимо разбить на криволинейные трапеции таким образом, чтобы отрезок [x0..x3] был разбит на равные части.

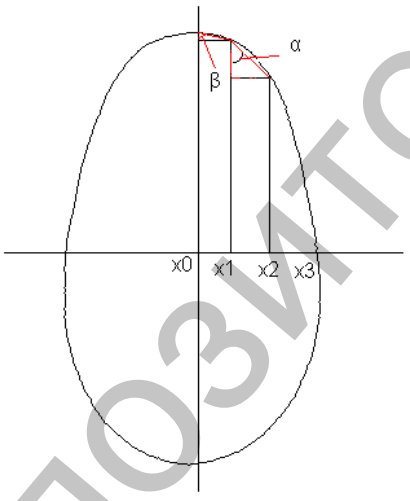
Через точки C и D провести прямую AB до пересечения с осями. Необходимо найти длину отрезка AB для всех изображений из папки. Результаты оформить в виде таблицы.



Лабораторная работа №6.

Даны контуры объектов. Правую верхнюю четверть необходимо разбить на криволинейные трапеции таким образом, чтобы отрезок [x0..x3] был разбит на равные части.

Необходимо найти отношение углов α/β для всех изображений из папки.



Также необходимо найти:

1. Степень отклонения точек контура от окружности.
2. Сравнение точек контура в полярной системе координат.

Лабораторная работа №7.

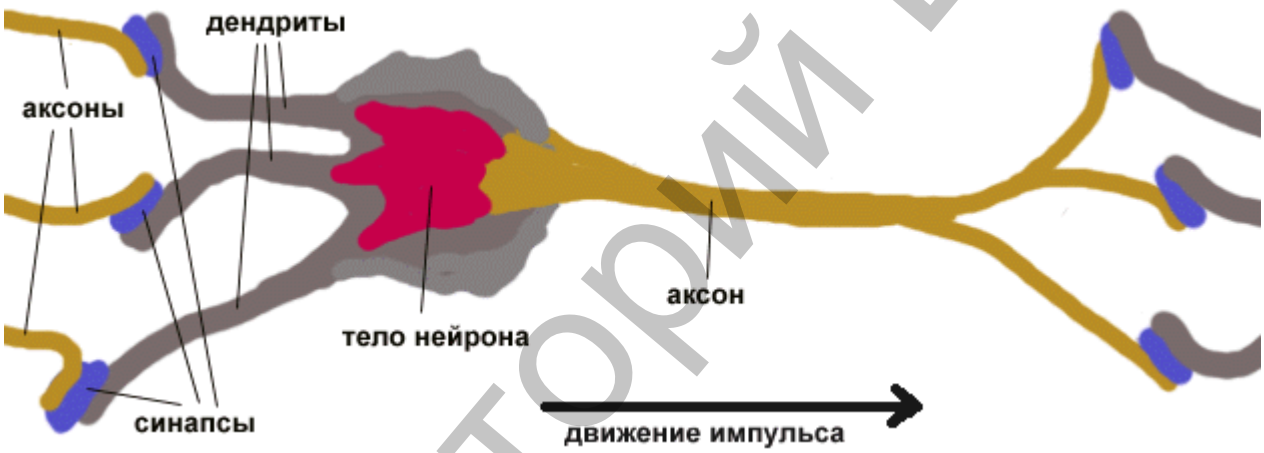
Задание.

Разработать нейронную сеть, используя результаты предыдущих лабораторных работ. С помощью нейронной сети решить задачу распознавания.

Указание к лабораторной работе.

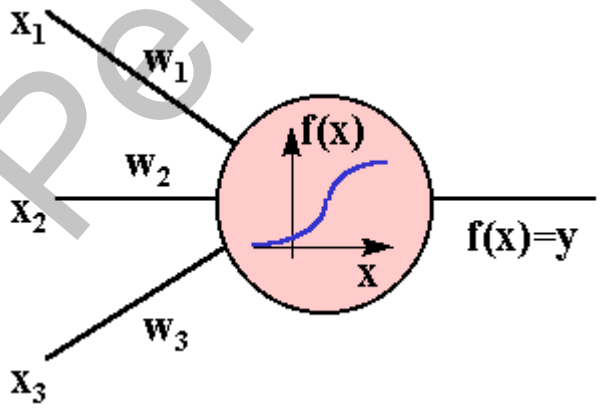
Нервная система и мозг человека состоят из нейронов, соединенных между собой нервными волокнами. Нервные волокна способны передавать электрические импульсы между нейронами. Все процессы передачи раздражений от нашей кожи, ушей и глаз к мозгу, процессы мышления и управления действиями - все это реализовано в живом организме как передача электрических импульсов между нейронами.

Рассмотрим строение биологического нейрона. Каждый нейрон имеет отростки нервных волокон двух типов - дендриты, по которым принимаются импульсы, и единственный аксон, по которому нейрон может передавать импульс. Аксон контактирует с дендритами других нейронов через специальные образования - синапсы, которые влияют на силу импульса.



Можно считать, что при прохождении синапса сила импульса меняется в определенное число раз, которое мы будем называть весом синапса. Импульсы, поступившие к нейрону одновременно по нескольким дендритам, суммируются. Если суммарный импульс превышает некоторый порог, нейрон возбуждается, формирует собственный импульс и передает его далее по аксону. Важно отметить, что веса синапсов могут изменяться со временем, а значит, меняется и поведение соответствующего нейрона.

Нетрудно построить математическую модель описанного процесса.



На рисунке изображена модель нейрона с тремя входами (дендритами), причем синапсы этих дендритов имеют веса w_1, w_2, w_3. Пусть к синапсам по-

ступают импульсы силы x_1, x_2, x_3 соответственно, тогда после прохождения синапсов и дендритов к нейрону поступают импульсы w_1x_1, w_2x_2, w_3x_3. Нейрон преобразует полученный суммарный импульс $x=w_1x_1+ w_2x_2+ w_3x_3$ в соответствии с некоторой передаточной функцией f(x). Сила выходного импульса равна $y=f(x)=f(w_1x_1+ w_2x_2+ w_3x_3)$.

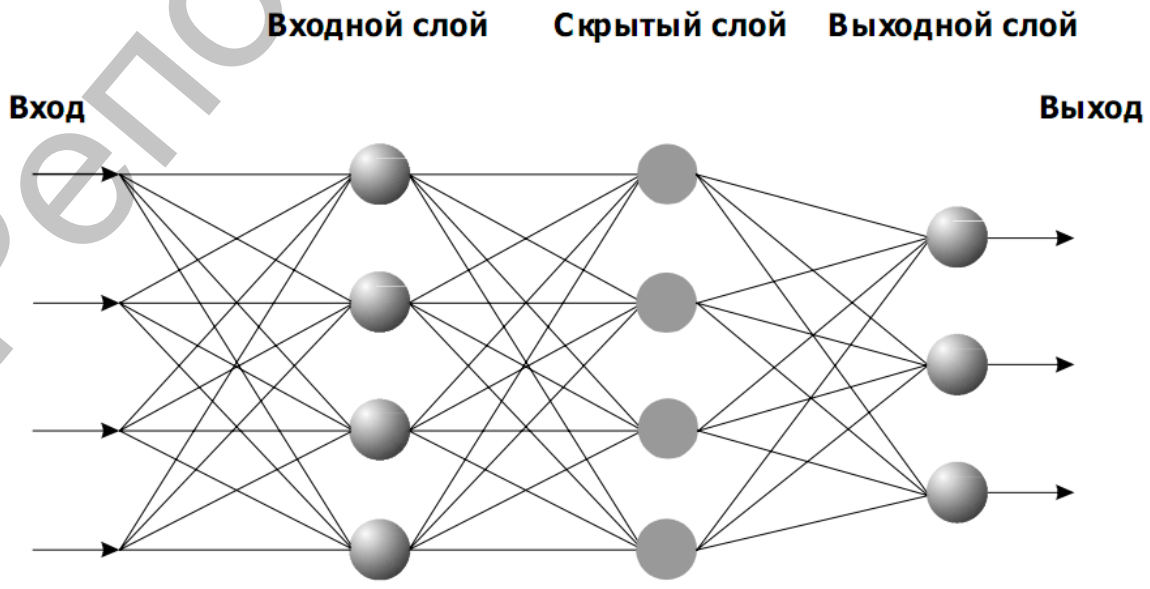
Таким образом, нейрон полностью описывается своими весами w_k и передаточной функцией f(x). Получив набор чисел (вектор) x_k в качестве входов, нейрон выдает некоторое число y на выходе.

1. Многослойный перцептрон.

Многослойная НС или многослойный перцептрон впервые были предложены и исследованы в 60-х годах Розенблаттом, Минским, Пейпертом и др. Но в дальнейшем исследования в данной области практически прекратились вследствие отсутствия эффективного алгоритма их обучения. Лишь в середине 1980-х несколькими исследователями независимо друг от друга был предложен алгоритм обучения многослойных персептронов, основанный на вычислении градиента функции ошибки.

Алгоритм был назван "обратным распространением ошибки". Стоить заметить, что в настоящее время алгоритм обратного распространения применяется конкретно для многослойных НС с последовательными связями.

На рисунке изображена укрупненная схема трехслойной НС, где нейроны помечены кружками, а связи между ними – стрелками. Для многослойных НС с последовательными связями характерно, что нейроны в таких сетях делятся на определенные группы с общим входным сигналом – слои, причем на каждый нейрон текущего слоя подаются все выходные сигналы с предыдущего слоя. В случае, если текущий слой является первым, то на каждый нейрон в его составе подаются все элементы внешнего входного сигнала.



Нейроны выполняют взвешенное суммирование элементов входных сигналов. К сумме элементов входных сигналов, домноженных на соответствующие синаптические веса, прибавляется смещение нейрона. Над результатом суммирования выполняется нелинейное преобразование - функция активации (передаточная функция). Значение функции активации есть выход нейрона.

Многослойный перцептрон — нейронная сеть, состоящая из слоев, каждый из которых состоит из элементов — нейронов (точнее их моделей). Эти элементы бывают трех типов: сенсорные (входные, S), ассоциативные (обучаемые «скрытые» слои, A) и реагирующие (выходные, R). Многослойным этот тип перцептронов называется не потому, что состоит из нескольких слоев, ведь входной и выходной слои можно вообще не оформлять в коде, а потому, что содержит несколько (обычно, не более двух — трех) обучаемых (A) слоев.

Модель нейрона— это элемент сети, который имеет несколько входов, каждый из которых имеет вес. Нейрон, получая сигнал, умножает сигналы на веса и суммирует получившиеся величины, после чего передает результат к другому нейрону или на выход сети. Здесь тоже многослойный перцептрон имеет отличия. Его функция — сигмоид, она выдает значения на промежутке от 0 до 1. К сигмоидам относится несколько функций:

Функция Ферми (экспоненциальная сигмоида): $$f(s) = \frac{1}{1 + e^{-2\alpha s}}$$

Рациональная сигмоида: $$f(s) = \frac{s}{|s| + \alpha}$$

Гиперболический тангенс: $$f(s) = th\frac{s}{\alpha} = \frac{e^{\frac{s}{\alpha}} - e^{-\frac{s}{\alpha}}}{e^{\frac{s}{\alpha}} + e^{-\frac{s}{\alpha}}}$$

где s — выход сумматора нейрона, α — произвольная константа.

Алгоритм метода обратного распространения ошибки:

1. Инициализация сети.

Весовым коэффициентам и смещениям сети присваиваются малые случайные значения из диапазонов.

2. Определение элемента обучающей выборки.

Текущие входы (X_0,X_1,…,X_{N-1}), должны различаться для всех элементов обучающей выборки. При использовании многослойного персептрона в качестве классификатора желаемый выходной сигнал (D_0,D_1,…,D_{N-1}) состоит из нулей за исключением одного единичного элемента, соответствующего классу, к которому принадлежит текущий входной сигнал.

3. *Вычисление текущего выходного сигнала.*

Текущий выходной сигнал определяется в соответствии с традиционной схемой функционирования многослойной нейронной сети.

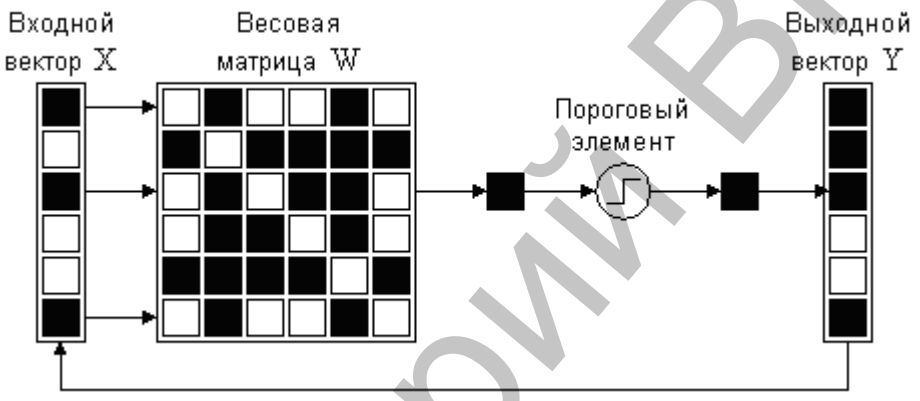
4. Настройка синаптических весов.

Синаптические веса настраиваются в соответствии с правилами метода обратного распространения, начиная с выходных нейронов сети, а затем проходя сеть в обратном направлении до первого слоя.

2. Нейронная сеть Хопфилда.

2. 1. Идея рекуррентности.

Нейронная сеть Хопфилда – это пример сети, которую можно определить как динамическую систему, у которой выход одной полностью прямой операции служит входом следующей операции сети, как показано на рисунке:



Сети, которые работают как системы обратной связи, называются “рекуррентными сетями”. Каждая прямая операция сети называется итерацией. Рекуррентные сети, подобно любым другим нелинейным динамическим системам, способны проявлять целое разнообразие различных поведений. В частности, один возможный образец поведения – это то, что система может быть устойчивой, т.е. она может сходиться к единственной фиксированной (неподвижной) точке. Когда неподвижная точка является входом в такую динамическую систему, то на выходе будем иметь ту же самую точку. Таким образом система остается зафиксированной в том же самом состоянии. Возможны периодические циклы или хаотическое поведение.

Было показано, что сети Хопфилда устойчивы. В общем случай может быть более одной фиксированной точки. То, к такой фиксированной точке будет сходиться сеть, зависит от исходной точки, выбранной для начальной итерации.

Неподвижные точки называются аттракторами. Множество точек (векторов), которые притягиваются к определенному аттрактору в процессе итераций сети, называется “областью притяжения” этого аттрактора. Множество неподвижных точек сети Хопфилда – это ее память. В этом случае сеть может действовать как ассоциативная память. Те входные векторы, которые попадают в сферу притяжения отдельного аттрактора, являются связанными (ассоциированными) с ним.

Например, аттрактор может быть некоторым желаемым образом. Область притяжения может состоять из зашумленных или неполных версий этого об-

раза. Есть надежда, что образы, которые смутно напоминают желаемый образ будут вспомнены сетью как ассоциированные с данным образом.

2.2. Бинарные сети Хопфилда.

На рисунке изображена бинарная сеть Хопфилда. Входные и выходные векторы состоят из “–1” и “+1” (вместо “–1” ,может быть использовано “0”) Имеется симметричная весовая матрица, состоящая из целых чисел с нулями (или “–1”) по диагонали $W = \|w_{ij}\|$. Входной вектор X умножается на весовую матрицу, используя обычное матрично-векторное умножение. Однако только 1 компонента выходного вектора Y используется на каждой итерации. Это процедура известна как “асинхронная коррекция”. Эта компонента, которая может быть выбрана случайно (подается на пороговый элемент, чей выход или –1, или +1). Соответствующая компонента входного вектора заменяется на это значение и, таким образом, образует входной вектор для следующей итерации. Процесс продолжается до тех пор, пока входные и выходные вектора не станут одинаковыми (то есть пока не будет достигнута неподвижная точка)

2.3. Описание алгоритма.

- Вычисляются компоненты выходного вектора y_j, j=1,2,..,n, по формуле

$$y_j = T\left(\sum_{i=1}^{n} w_{ij} x_i\right) \quad (1)$$

где $T(x) = \begin{cases} -1, & \text{если } x < 0 \\ 1, & \text{если } x > 0 \end{cases}$

Выполнить асинхронную коррекцию, т.е.

1. начать с входного вектора $(x_1, x_2, \ldots, x_n)$;
2. найти y_j согласно формуле (1);
3. заменить $(x_1, x_2, \ldots, x_n)$ на $(y_1, x_2, x_3, \ldots, x_n)$ и подать Y на вход X.
4. повторить процесс, чтобы найти y_2, y_3 и т.д. (y_j могут выбираться случайно)

- Повторять шаги 2-3 до тех пор, пока вектор $Y = (y_1, y_2, \ldots, y_n)$ не перестанет изменяться. Каждый шаг уменьшает величину энергии связей

$$E = \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} w_{ij} x_i x_j \quad (2)$$

Так, что обеспечивается сходимость к неподвижной точке (аттрактору).

Асинхронная коррекция и нули на диагонали матрицы W гарантируют, что энергетическая функция (2) будет уменьшаться с каждой итерацией. Асинхронная коррекция – это особенно существенно для обеспечения сходимости к неподвижной точке. Если мы допустим, чтобы весь вектор корректировался на каждой итерации, то можно получить сеть с периодическими циклами как терминальными состояниями аттакторов, а не с неподвижными точками.

2.4. Образцы поведения.

Именно весовая матрица отличает поведение одной сети Хопфилда от другой, так что возникает вопрос: “Как определить эту весовую матрицу?”

Ответ – надо задать определенные весовые вектора, которые называют экземплярами. Есть надежда, что эти экземпляры будут фиксированными точками результирующей сети Хопфилда. Хотя это не всегда так. Для того, чтобы экземпляры были аттракторами, весовую матрицу $W = \|w_{ij}\|$ надо задать таким образом:

$$w_{ij} = \begin{cases} \sum_{k=1}^{N} (x_{k_i} - 1)(x_{k_j} - 1), \text{для } i \neq j \\ 0, \text{для } i = j \end{cases} \quad (3)$$

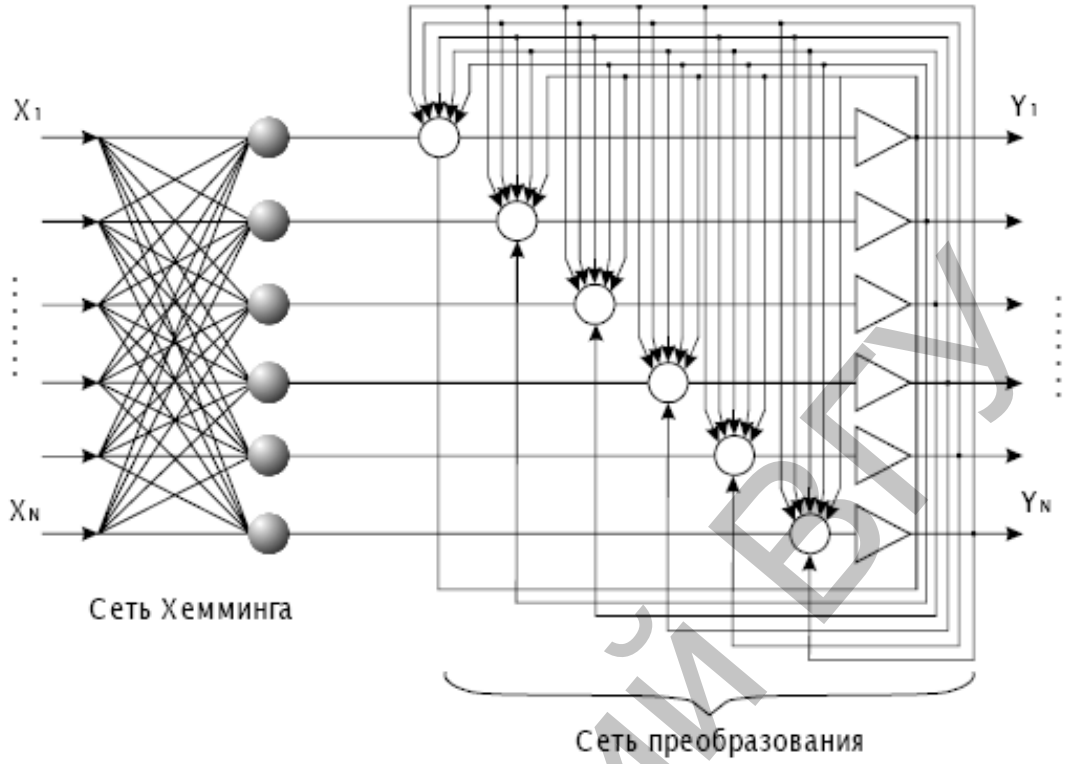
где N – число заданных экземпляров, а $X_k = (x_{k_1}, x_{k_2}, \ldots, x_{k_n})$ – k-ый экземпляр.

Если экземпляры векторов образуют множество ортогональных векторов, то можно гарантировать, что если весовая матрица выбирается как показано выше, то каждый экземпляр вектора будет неподвижной точкой. Однако в общем случае для того, чтобы экземпляры приводили к неподвижным точкам, ортогональность не обязательна.

3. Нейронная сеть Хемминга.

Расстояние Хемминга между двумя бинарными векторами одинаковой длины – это число несовпадающих бит в этих векторах. Нейронная сеть, которая реализует параллельное вычисление расстояний Хемминга от входного вектора до нескольких векторов-образцов, носит название сети Хемминга.

НС Хемминга – это двухслойная прямонаправленная сеть, способная распознавать объект из изображения с шумами, заложенный в нее на стадии обучения сети.

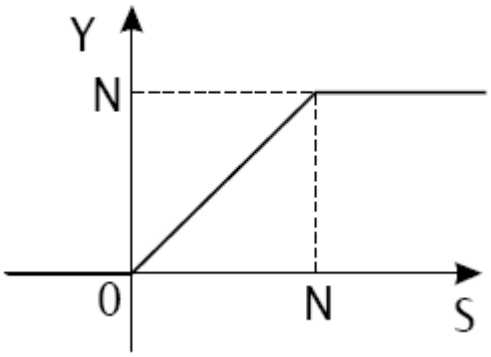


На рисунке первый слой, называемый подсетью или сетью Хемминга, вычисляет расстояние Хемминга между входным вектором и вектором, занесенным в память путем изменения значений синаптических весов нейронов сети. Подсеть или сеть преобразования выполняет функцию выбора нейрона из первого слоя, выходной сигнал которого имеет наибольшую амплитуду, то есть нейрон, для которого расстояние Хемминга между входным вектором и вектором-образцом является наименьшим. Такого
рода нейрон также называется нейроном-победителем. Настройка синаптических весов и смещений в рассмотренной сети, осуществляемая
путем вычисления расстояния Хемминга, определяется посредством формулы

$$w_{i,j} = \frac{x_i^j}{2}, b_j = \frac{N}{2}, 1 \le i \le N, 1 \le j \le M$$

где b – смещение, N – размерность входного сигнала, M – количество нейронов в сети.

Вид передаточной функции сети Хемминга приведен на рисунке. Как видно из рисунка она представляет собой линейную функцию с насыщением.



Функция, по которой производится непосредственно функционирование сети Хемминга, представлена в формуле

$$y_j = \left(\sum_{i=1}^{N} w_{i,j} x_i - b_j \right), 1 \le j \le M$$

где y_j – выход j- го нейрона сети.

Сеть Хемминга может распознавать только слабозашумленные входные сигналы, что также ограничивает возможность ее применения в задаче распознавания изображений. Однако сама возможность использования данной сети для распознавания зашумленных изображений, а также возможность несовпадения размерностей входного и выходного сигналов сети делает ее использование более выгодным, чем, например, использование
сети Хопфилда в задачах распознавания. Также стоит упомянуть о высоком быстродействии сети вследствие того, что выходной сигнал формируется в результате прохода всего лишь одного слоя нейронов. Это важно рассматривать в сравнении, например, с сетями циклического функционирования, когда сигнал многократно проходит через нейроны в слоях сети, и количество слоев, таким образом, в большой степени определяет быстродействие сети в целом. Обучение сети реализуется на базе
довольно простого алгоритма, основанного на вычислении расстояния Хемминга, что также определяет высокую скорость обучения сети. В отличие от сети Хопфилда, емкость сети Хемминга не зависит от размерности входного сигнала, она в точности равна количеству нейронов (M).