КОМПЬЮТЕРНЫЕ НАУКИ 2018. Т. 28. Вып. 4

УДК 519.17, 510.5

© *А. Ю. Сапаров*

ВОССТАНОВЛЕНИЕ ПОСЛЕДОВАТЕЛЬНОСТИ ЗАПИСИ В СКАНИРОВАННЫХ РУКОПИСНЫХ ТЕКСТАХ

Рассматривается задача распознавания рукописных текстов с растровых изображений. Описывается метод восстановления последовательности записи рукописного текста, который позволит свести задачу offline-распознавания к задаче online-распознавания. Метод заключается в поиске эйлерова пути с минимальным весом в графе скелета рукописных символов. В качестве весов рассматриваются некоторые числовые характеристики, отражающие сложность перехода из одного ребра в другое через общую вершину. Для этого строится таблица всевозможных комбинаций пар. При отсутствии в исходном графе эйлерова пути выполняется поиск пути с минимальным числом разрывов. Для разбиения ребер на пары и вычисления весов в вершинах нечетной кратности вводится понятие виртуального ребра, переход по которому означает образование разрыва в пути. Рассматривается алгоритм поиска пути в скелете символа, основанный на алгоритме Флери поиска эйлерова пути.

Ключевые слова: граф скелета рукописного символа, путь в скелете, виртуальное ребро.

DOI: 10.20537/vm180411

Введение

Основное отличие в распознавании сканированных текстов от online-распознавания заключается в различии входных данных [1–3]. При online-распознавании основным критерием при принятии решений является анализ местоположения пишущего устройства в каждый момент времени. Таким образом, в этом случае тексты представлены в виде некоторого упорядоченного набора координат на плоскости: $T = ((x_1, y_1), (x_2, y_2), \ldots, (x_k, y_k))$. Для сканированных текстов местоположение пишущего устройства неизвестно, и такие изображения текстов представлены в виде прямоугольной матрицы: $A = (a_{i,j})_{i=1,j=1}^{n,m}$, где каждому элементу матрицы соответствует точка (пиксель) растрового изображения.

Алгоритмы online-распознавания достигли практически 100 %-ной точности, чего нельзя сказать о распознавании текстов со сканированных изображений. Большинство алгоритмов offline-распознавания основываются на нейронных сетях [4,5] и структурном анализе [6,7]. Также рассматриваются методы, основанные на анализе дескрипторов функций длины хорды [8] и применении вейвлет-преобразования Хаара [9]. Из выводов рассмотренных работ точность распознавания достигается от 85 % до 95 %, но это касается только отдельных символов (в большинстве случаев только цифр). При анализе сплошных текстов, содержащих большее число всевозможных символов, точность распознавания падает. В таких документах к дополнительной сложности приводит необходимость предварительной сегментации исходного изображения, для чего могут быть использованы алгоритмы множественной сегментации и выбора наилучшей комбинации сегментов [10, 11]. Но в текстах со сложной структурой число способов сегментации может быть довольно большим, что приводит к заметному увеличению сложности алгоритма и времени его работы. Если рассматривать математические или другие сложноструктурированные тексты, то говорить о высокой точности пока не приходится. Для анализа сплошных и сложноструктурированных текстов требуется рассмотреть новые подходы. Предлагается использовать методы online-распознавания при решении задачи offline-распознавания.

Чтобы применить к сканированным изображениям алгоритмы online-распознавания [12–15], необходимо решить задачу получения последовательности T из матрицы A:

$$F\colon (a_{i,j})_{i=1,j=1}^{n,m} \to ((x_1, y_1), (x_2, y_2), \ldots, (x_k, y_k)).$$

Решение данной задачи позволит увеличить точность распознавания рукописных символов со сканированных изображений.

§ 1. Описание задачи и основные обозначения

Задача online-распознавания решается анализом движения пишущего устройства по специальному экрану и сопоставления полученных данных с некоторыми шаблонами. Таким образом, основным параметром в алгоритмах online-распознавания является последовательность координат пишущего устройства. В сканированных изображениях получение последовательности координат не представляется возможным. Пусть исходное изображение представлено в виде прямоугольной матрицы M, где $M_{i,j} = 1$, если точка (i, j) принадлежит фрагменту текста, и $M_{i,j} = 0$, если точка (i, j) принадлежит фону.

Определение 1. Пусть $A = (a_{i,j})$ — целочисленная матрица. Элементы $a_{i-1,j-1}$, $a_{i-1,j}$, $a_{i-1,j+1}$, $a_{i,j-1}$, $a_{i,j+1}$, $a_{i+1,j-1}$, $a_{i+1,j}$, $a_{i+1,j+1}$ будем называть *соседними* относительно элемента $a_{i,j}$.

Найдем *скелет* M_s исходного изображения, воспользовавшись одним из алгоритмов «утончения» [16]. Тогда в M_s толщина каждой рукописной кривой будет равняться 1 точке.

Определение 2. Пусть матрица A_s является скелетом. Будем использовать следующие обозначения:

- $a_{i,j} = 1$ — *крайняя точка*, если у $a_{i,j}$ ровно 1 ненулевой сосед;
- $a_{i,j} = 1$ — *точка пересечения*, если у $a_{i,j}$ более 2 ненулевых соседей;
- $a_{i,j} = 1$ — *изолированная точка*, если у $a_{i,j}$ нет ненулевых соседей;
- $a_{i,j} = 1$ — *промежуточная точка*, если у $a_{i,j}$ ровно 2 ненулевых соседа;
- $a_{i,j} = 0$ — *точка фона*.

Определение 3. Крайние точки и точки пересечения будем называть *критическими* точками.

Определение 4. Две промежуточные точки a_1 и a_2 будем называть *смежными*, если они являются соседними (либо совпадают), либо существует промежуточная точка a_3 такая, что a_1 — сосед a_3, а a_1 и a_2 — *смежные*. Иными словами, смежными считаются все пары промежуточных точек, связанные друг с другом через другие промежуточные точки.

Определение 5. Точку пересечения будем называть *составной*, если хотя бы одна из соседних точек также является точкой пересечения. Термин «*составная* точка пересечения» также будем применять ко всему набору соседних точек пересечения. Это такие точки, которые образуются при невозможности выделения в скелете *простой* (состоящей из одного пикселя) точки пересечения.

Определение 6. Две критические точки будем называть *смежными*, если они имеют *смежных* соседей (промежуточные точки). Иными словами, критические точки считаются смежными, если они связаны через промежуточные точки.

Для построения математической модели, отражающей структуру изображения рукописного текста, будем использовать теорию графов. Это позволит как сохранить с необходимой точностью исходную структуру текста [19, 20], так и уменьшить размерность изображения. Представим скелет изображения рукописного текста в виде неориентированного графа $G = (S, U)$ [18], где S — множество вершин, а U — множество ребер. Вершинам S будут соответствовать критические точки, а ребрам U — пары смежных критических точек.

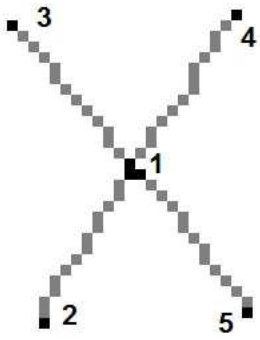

Рис. 1. Пример скелета рукописного символа x

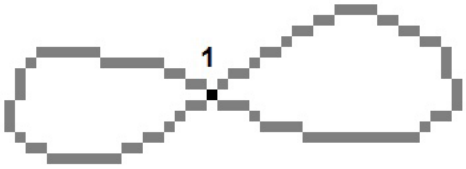

Рис. 2. Пример скелета рукописного символа ∞

Пример 1. Рассмотрим пример скелета изображения рукописного символа x (рис. 1). На скелете выделены четыре крайние точки, условно обозначенные цифрами 2–5, и одна составная точка пересечения, обозначенная цифрой 1. Как видно из рисунка, при удалении хотя бы одной из точек уже нарушается связность компонент, поэтому точка пересечения будет составной.

В результате полученный граф будет выглядеть следующим образом: $G = (\{1,2,3,4,5\}, \{(1,2),(1,3),(1,4),(1,5)\})$.

Пример 2. Рассмотрим пример скелета изображения рукописного символа ∞ (рис. 2). На скелете выделена одна точка пересечения, условно обозначенная цифрой 1. Граф символа будет выглядеть следующим образом: $G = (\{1\}, \{(1,1),(1,1)\})$. Из этого следует, что полученные графы могут содержать петли и кратные ребра. Пока кратные ребра будем обозначать с индексом: $G = (\{1\}, \{(1,1)_1, (1,1)_2\})$.

Для приведения задачи offline-распознавания к задаче online-распознавания необходимо восстановить последовательность записи всех ребер полученного графа. Необходимо найти такую последовательность $T = (u_{i_1}, u_{i_2}, \ldots, u_{i_n})$, где $\forall i\ u_i \in U$, которая содержит все ребра. Для примера 1 это 4 ребра. Каждое ребро по отдельности может быть записано как в одном направлении, так и в обратном, т. е. 16 или 2^m вариантов, где m — количество ребер. В свою очередь, запись этих 4 ребер может быть произведена в разной последовательности, т. е. 24 или $m!$ вариантов. В итоге всего $2^m \cdot m!$ вариантов записи. Отсюда следует, что для сложных скелетов количество всевозможных способов записи является довольно большим числом, и перебор всех вариантов является довольно трудоемкой задачей.

Из этого можно предположить следующее:

- для каждого ребра необходимо указывать направление движения;
- последовательность ребер необязательно должна быть непрерывной;
- число разрывов должно быть минимально.

Определение 7. Последовательность ребер $T = (u_{i_1}, u_{i_2}, \ldots, u_{i_n})$, где $\forall i\ u_i \in U$, которая содержит все ребра, будем называть *путем в скелете*. Отличается от пути в обычном понимании тем, что путь в скелете может содержать разрывы.

Очевидно, что в первом примере одним из наиболее оптимальных способов записи символа является следующий путь: $T = ((3,1),(1,5),(4,1),(1,2))$. Во втором примере путь будет выглядеть следующим образом: $T = ((1,1)_1,(1,1)_2)$. В данном случае необходимо указывать направление движения для каждого ребра.

Определение 8. *Графом скелета рукописного символа* будем называть граф $G = (S,U)$, в котором каждой вершине $s \in S$ соответствует критическая точка (возможно, составная точка пересечения) скелета изображения, а каждому ребру — набор точек скелета $(p_1,p_2,\ldots,p_{k-1},p_k)$, где p_1, p_k — смежные критические точки, а p_2–p_{k-1} — промежуточные точки, причем $\forall i$ p_i и p_{i+1} являются соседями. Иными словами, каждое ребро задается координатами всех точек кривой, соединяющих соответствующие вершины.

Определение 9. Так как каждое ребро u графа скелета задается упорядоченным набором точек, то для него задается некоторое направление. Ребро с противоположным направлением будем обозначать через $-u$.

Замечание 1. При обозначении отдельных ребер верны следующие равенства: $(s_1,s_2) = (s_2,s_1)$, где s_1, s_2 — вершины графа; $-u = u$, где u — ребро графа.

Замечание 2. При обозначении ребер в пути равенства в замечании 1 не выполняются, поэтому при обозначении таких ребер следует явно указывать направление движения. Заметим, что (s_1,s_2) — это только сокращенное обозначение ребра. В полном виде ребро обозначается как $(s_1,p_2,\ldots,p_{k-1},s_2)$, где $p_2,\ldots,p_{k-1}$ — непрерывная последовательность промежуточных точек, соединяющих критические точки s_1 и s_2.

Определение 10. *Началом ребра* $u = (p_1,\ldots,p_k)$ будем называть первую точку направления: $s(u) = p_1$.

Определение 11. *Концом ребра* $u = (p_1,\ldots,p_k)$ будем называть последнюю точку направления: $e(u) = p_k$.

Замечание 3. Заметим, что $s(-u) = e(u)$ и $e(-u) = s(u)$. Если $s(u) = e(u)$, то u — петля.

Оптимальным решением задачи является эйлеров путь [17] в графе с наименьшим весом, если он существует. Поиск пути будет осуществляться на основе сложности перехода по каждой вершине. Для каждой пары (u_i,u_{i+1}) из T ставится в соответствие некоторый числовой вес, отражающий сложность перехода из ребра u_i в ребро u_{i+1} через общую вершину. Например, если (u_i,u_{i+1}) вблизи общей вершины представлена прямой линией, то вес этой пары равен 1; если представлена ломаной, то вес пары равен 2. Если кривой не существует, то вес равен 0. Каждая пара, которая соответствует отсутствующей кривой, есть момент отрыва пишущего устройства от листа бумаги. Нулевой вес объясняется не только отсутствием необходимости траты чернил, но и тем, что такой переход может быть произведен намного быстрее, чем при выполнении записи. Но тогда это можно понимать как возможность мгновенного перехода в любую точку. Учитывая, что граф ограничен отдельным символом либо компонентой связности, под такими переходами будут подразумеваться только переходы на сравнительно небольшие расстояния. В общем виде должны быть выполнены следующие условия [21]:

$$\begin{cases} |\{i : e(u_i) \neq s(u_{i+1})\}| \to \min, \\ \sum_i w((u_i,u_{i+1})) \to \min, \end{cases} \tag{1.1}$$

где $e(u)$ и $s(u)$ — конец и начало ребра u соответственно, а $w((u,v))$ — вес пары (u,v). Таким образом, необходимо найти такой путь, в котором число разрывов минимально и минимальна сумма весов всех входящих в путь пар. Приоритет дается первому условию, поэтому путь, составленный чередованием существующих и несуществующих кривых, не может быть решением задачи.

§ 2. Описание алгоритма

Если граф $G = (S, U)$ является связным, то условием существования эйлерова пути, по теореме Эйлера [17], является наличие в нем не более 2 вершин нечетной кратности. В таком случае, чтобы выполнилось первое условие в (1.1), достаточно выбрать эйлеров путь в графе G. В обычном понимании весом эйлерова пути является сумма весов его ребер. С учетом того, что эйлеров путь содержит все ребра графа, и при том по одному разу, веса всех путей должны быть равны. Но в данном случае в качестве весов рассматриваются не какие-то числовые параметры отдельных ребер, а параметры, отражающие сложность перехода от одного ребра в другое. В таком случае веса разных эйлеровых путей будут различными. Наиболее простым для понимания является алгоритм перебора всех эйлеровых путей и выбора среди них пути с минимальным весом, чтобы выполнилось второе условие в (1.1).

Для поиска эйлерова пути можно воспользоваться алгоритмом Флери [17]. Начинать с некоторой вершины p (если в графе есть вершины нечетной кратности, то нужно начинать с них) и каждый раз вычеркивать пройденное ребро. Не проходить по ребру, если удаление этого ребра приводит к разбиению графа на две связные компоненты (не считая изолированных вершин).

Если эйлеров путь в графе не существует, то рассматривается модифицированная задача поиска эйлерова пути, которая разрешает иметь в итоговом результате минимально возможное количество разрывов. Для этого можно воспользоваться алгоритмом Флери [17] с некоторыми модификациями, которые заключаются в том, что алгоритм может быть применен к графу с любым числом вершин нечетной кратности, и, при невозможности дальнейшего движения, из одной вершины нечетной кратности можно перейти в другую.

Вернемся к примеру 1. В данном случае граф содержит 5 вершин, 4 из которых кратности 1. Из этого следует, что эйлерова пути не существует. Опишем действия, которые нужно выполнить для нахождения пути с наименьшим весом. Очевидно, что путь должен состоять из 2 непрерывных фрагментов, например $((3,1),(1,5))$ и $((4,1),(1,2))$ или $((3,1),(1,4))$ и $((2,1),(1,5))$ и т. д. Построим таблицу весов:

u \ v	$(3,1)$	$(1,5)$	$(4,1)$	$(1,2)$
$(3,1)$	0	1	0	2
$(1,5)$	0	0	0	0
$(4,1)$	0	2	0	1
$(1,2)$	0	0	0	0

Или с учетом противоположных направлений ребер:

u \ v	$(1,2)$	$(1,3)$	$(1,4)$	$(1,5)$	$(2,1)$	$(3,1)$	$(4,1)$	$(5,1)$
$(1,2)$	0	0	0	0	3	0	0	0
$(1,3)$	0	0	0	0	0	3	0	0
$(1,4)$	0	0	0	0	0	0	3	0
$(1,5)$	0	0	0	0	0	0	0	3
$(2,1)$	3	2	1	2	0	0	0	0
$(3,1)$	2	3	2	1	0	0	0	0
$(4,1)$	1	2	3	2	0	0	0	0
$(5,1)$	2	1	2	3	0	0	0	0

Таблица задает функцию весов $w(u, v)$. Например, вес перехода из ребра $u = (3,1)$ в ребро $v = (1,5)$ указан на пересечении строки $(3,1)$ с колонкой $(1,5)$ и равен 1. Для простоты веса считаются по следующему правилу:

- если $e(u) \neq s(v)$, тогда $w(u, v) = 0$;

- если $u = -v$, тогда $w(u, v) = 3$;
- если $e(u) = s(v)$ и кривая (u, v) в точке $s(v)$ меняет направление (имеется угол более 45 градусов), тогда $w(u, v) = 2$;
- если $e(u) = s(v)$ и кривая (u, v) в точке $s(v)$ не меняет направления, тогда $w(u, v) = 1$.

Начнем поиск пути с вершины 3 как одной из вершин нечетной кратности. Из вершины 3 выходит единственное ребро $(3, 1)$, поэтому проходим по этому ребру и вычеркиваем его. Из вершины 1 выходят сразу 3 ребра: $(1, 4)$, $(1, 5)$ и $(1, 2)$. При вычеркивании этих ребер не нарушается связность графа, поэтому переход может быть произведен в любом направлении. По принципу минимального веса выбираем ребро $(1, 5)$. Из вершины 5 после вычеркивания ребра $(1, 5)$ больше не выходит ни одно ребро, т. е. вершина 5 становится изолированной, поэтому выбираем из оставшихся вершин нечетной кратности 2 и 4. Выбираем вершину 4. Из этой вершины выходит единственное ребро $(4, 1)$, поэтому движемся по нему и вычеркиваем его. После этого в графе остается единственное ребро $(1, 2)$. Выбираем его, и поиск пути завершается. В итоге получен следующий путь: $T = ((3, 1), (1, 5), (4, 1), (1, 2))$, который имеет минимально возможное число разрывов и минимальный вес.

Стоит отметить, что итоговый путь зависит от выбора первоначальной вершины и выбора вершин после попадания на изолированные вершины. В этих случаях выбор вершины может осуществляться по правилу записи текста слева направо, т. е. выбираем самую левую свободную вершину нечетной кратности. Учитывая, что при online-распознавании в качестве шаблонов рассматриваются сразу несколько способов записи, то нет необходимости нахождения единственно верного пути записи. Но с целью увеличения точности распознавания рекомендуется находить сразу несколько путей с минимальными весами. Также стоит отметить, что в рассмотренном примере веса оставшихся переходов не зависят от выполненных ранее переходов. В реальности каждый выбор перехода должен осуществляться с учетом возможностей выбора в дальнейшем.

Замечание 4. Из постановки задачи, а именно из условия 1 в (1.1), следует, что путь не должен содержать разрывов в вершинах четной кратности и должен содержать не более одного разрыва в каждой вершине нечетной кратности.

Пусть k — кратность вершины s графа G, и пусть k — четное число. Из замечания 4 следует, что при поиске пути в вершину s должны попасть $\frac{k}{2}$ раз. Другими словами, при каждом попадании в вершину s должна вычеркиваться пара ребер, выходящих из этой вершины. Необходимо найти такой набор пар ребер, чтобы сумма весов переходов была минимальной. Количество всевозможных пар ребер равно $(k-1)\cdot(k-3)\cdot\ldots\cdot 1$. При $k = 6$ количество равно $5 \cdot 3 \cdot 1 = 15$. Учитывая особенности рукописных символов, можно сделать выводы, что большинство точек пересечения будут кратности 4. Вершины кратности более 6 практически невозможны. Следовательно, выбор алгоритма поиска комбинации пар с минимальным весом не сильно влияет на сложность всего алгоритма поиска пути.

Определение 12. Пусть граф $G = (S, U)$ содержит вершину $s \in S$ кратности k, где k — четное число. Множество всех ребер, выходящих из s, обозначим через $U(s)$. Всевозможную комбинацию пар ребер из $U(s)$ обозначим через $U^2(s)$. Вес минимальной комбинации обозначим через $W_{\min}(s)$.

Рассмотрим вершину 1 из примера 1. Кратность вершины равна 4. Возможные комбинации пар $(U^2(1))$:

$$((2, 1), (1, 4)), ((3, 1), (1, 5));$$
$$((2, 1), (1, 3)), ((4, 1), (1, 5));$$
$$((2, 1), (1, 5)), ((3, 1), (1, 4)).$$

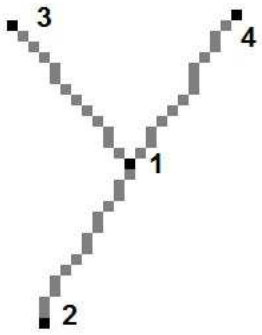

Рис. 3. Пример точки пересечения нечетной кратности

Веса пар при этом равны:

$$w((2,1),(1,4)) = 1, \quad w((3,1),(1,5)) = 1;$$
$$w((2,1),(1,3)) = 2, \quad w((4,1),(1,5)) = 2;$$
$$w((2,1),(1,5)) = 2, \quad w((3,1),(1,4)) = 2.$$

Минимальную сумму при этом имеет первый набор. Следовательно, при попадании в вершину 1 из ребра $(2,1)$ дальше необходимо двигаться по ребру $(1,4)$, а при попадании в вершину 1 из ребра $(3,1)$ — по ребру $(1,5)$.

Определение 13. Пусть в графе $G = (S,U)$ s — вершина нечетной кратности. Ребро $(s,0)$, назовем *виртуальным*, если $0 \notin S$. Вершину 0 будем называть *виртуальной* вершиной.

Замечание 5. Вес перехода из любого ребра в виртуальное равен 0.

Пусть теперь кратность вершины s нечетная, и разбиение по парам невозможно. Рассмотрим пример 3. Вершина 3 имеет кратность, равную 3. Добавим виртуальное ребро $(s,0)$, тогда комбинации пар будут выглядеть следующим образом:

$$((2,1),(1,4)),((3,1),(1,0));$$
$$((2,1),(1,3)),((4,1),(1,0));$$
$$((2,1),(1,0)),((3,1),(1,4)).$$

Веса пар при этом равны:

$$w((2,1),(1,4)) = 1, \quad w((3,1),(1,0)) = 0;$$
$$w((2,1),(1,3)) = 2, \quad w((4,1),(1,0)) = 0;$$
$$w((2,1),(1,0)) = 0, \quad w((3,1),(1,4)) = 2.$$

Минимальную сумму при этом имеет первый набор. Следовательно, при попадании в вершину 1 из ребра $(2,1)$ дальше необходимо двигаться по ребру $(1,4)$, а при попадании в вершину 1 из ребра $(3,1)$ — по виртуальному ребру, т. е. прервать непрерывную запись. Таким образом, виртуальное ребро необходимо только для разбиения исходного множества ребер по парам, и оно определяет точку разрыва в записи.

В результате использования комбинации пар ребер и поиска среди них комбинации с минимальной суммой весов, алгоритм Флери позволяет находить в графе путь с наименьшим общим весом. Виртуальное ребро позволяет решить проблему разбиения по парам нечетного числа ребер.

Замечание 6. В алгоритме Флери в каждый момент времени анализируется состояние только текущей вершины, следовательно, выбор пути с минимальным весом должен основываться только на весах текущей вершины.

Определение 14. Пусть T — некоторый путь в графе $G = (S, U)$. Через $|T|$ будем обозначать количество элементов в T. Соответственно, $T_{|T|}$ будет последним элементом в T.

В общем виде алгоритм поиска в графе $G = (S, U)$ пути с минимальным весом выглядит следующим образом.

1. Создать пустой путь T.
2. Для каждой точки пересечения нечетной кратности в графе G добавить виртуальное ребро.
3. Для каждой точки пересечения $s \in S$ найти комбинации пар $U^2(s)$. Отсортировать $U^2(s)$ в порядке возрастания суммы весов.
4. Если граф не содержит вершин нечетной кратности (виртуальные ребра не учитываются), то выполнить пункт (а); иначе, выполнить пункт (б).
   (а) Выбрать самую левую вершину s_0 из общего числа вершин. В первой строке $U^2(s_0)$ найти пару $((s_2, s_0), (s_0, s_1))$ с максимальным весом.
   (б) Выбрать самую левую вершину s_0 нечетной кратности. Если s_0 — крайняя точка, то перейти к пункту 6; иначе, в первой строке $U^2(s_0)$ найти пару $((s_2, s_0), (s_0, s_1))$, где (s_2, s_0) — виртуальное ребро.
5. Добавить ребро (s_0, s_1) в T. Удалить ребро (s_0, s_1) из G. Если все ребра, кроме виртуальных, удалены, то поиск завершается. Обозначить вершину s_1 через s_0. Если s_0 после удаления ребра стала изолированной точкой, то перейти к пункту 4 (б).
6. Если s_0 — крайняя точка, то выполнить пункт (а); иначе, выполнить пункт (б).
   (а) Выбрать единственное ребро (s_0, s_1), выходящее из s_0.
   (б) В $U^2(s_0)$ найти первую пару $(T_{|T|}, (s_0, s_1))$, удовлетворяющую следующему условию: при удалении ребра (s_0, s_1) из G не нарушается связность G, либо связность нарушается, но в обеих частях содержатся вершины нечетной кратности. Если (s_0, s_1) — виртуальное ребро, то перейти к пункту 4.
7. Перейти к пункту 5.

§ 3. Пример графа с точками пересечения четной кратности

Рассмотрим пример скелета символа $\not\subset$ (рис. 4).

Выполним алгоритм поиска пути для графа этого символа. Исходный граф состоит из 6 вершин и 6 ребер: $G = (\{1, 2, 3, 4, 5, 6\}, \{(1, 2), (2, 3), (3, 4), (5, 3), (3, 2), (2, 6)\})$. Для удобства ребра обозначим через $7, 8, 9, 10, 11, 12$. Тогда граф примет вид $G = (\{1, 2, 3, 4, 5, 6\}, \{7, 8, 9, 10, 11, 12\})$.

1. Пусть $T = \emptyset$.
2. Граф G не содержит точек пересечения нечетной кратности.
3. В графе G две точки пересечения: 2 и 3.
   $U^2(2) = (7, 8)(w = 1), (11, 12)(w = 1), (7, 11)(w = 2)$,
   $(8, 12)(w = 2), (7, 12)(w = 2), (8, 11)(w = 2)$;
   $U^2(3) = (8, 9)(w = 1), (10, 11)(w = 1), (8, 11)(w = 2), (9, 10)(w = 2), (8, 10)(w = 2)$,
   $(9, 11)(w = 2)$, где $w = 1$ и $w = 2$ — веса переходов.

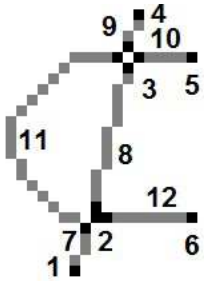

Рис. 4. Пример скелета рукописного символа $\not\subset$

4. В графе 4 вершины нечетной кратности. Поэтому выполняем пункт 4 (б).

4 (б) $s_0 = 1$ — крайняя точка. Переходим к пункту 6.

6. s_0 — крайняя точка, поэтому выполняем пункт 6 (а).

6 (а) $(s_0, s_1) = 7 = (1, 2)$.

7. Переходим к пункту 5.

5. Добавляем ребро 7 в T. $T = (7) = ((1, 2))$. Удаляем ребро 7 из G: $G = (\{1, 2, 3, 4, 5, 6\}, \{8, 9, 10, 11, 12\})$. Переходим в вершину 2: $s_0 = 2$. Вершина 2 не стала изолированной, поэтому переходим к следующему пункту.

6. $s_0 = 2$ — точка пересечения. Выполняем пункт 6 (б).

6 (б) В $U^2(2)$ первая пара — $(T_{|T|}, (s_0, s_1)) = (7, (s_0, s_1)) = (7, 8)$. При удалении ребра 8 связность графа G не нарушается.

7. Переходим к пункту 5.

5. Добавляем ребро 8 в T. $T = (7, 8) = ((1, 2), (2, 3))$. Удаляем ребро 8 из G: $G = (\{1, 2, 3, 4, 5, 6\}, \{9, 10, 11, 12\})$. Переходим в вершину 3: $s_0 = 3$. Вершина 3 не стала изолированной, поэтому переходим к следующему пункту.

6. $s_0 = 3$ — точка пересечения. Выполняем пункт 6 (б).

6 (б) В $U^2(3)$ первая пара — $(T_{|T|}, (s_0, s_1)) = (8, (s_0, s_1)) = (8, 9)$. При удалении ребра 9 связность графа G не нарушается.

7. Переходим к пункту 5.

5. Добавляем ребро 9 в T. $T = (7, 8, 9) = ((1, 2), (2, 3), (3, 4))$. Удаляем ребро 9 из G: $G = (\{1, 2, 3, 4, 5, 6\}, \{10, 11, 12\})$. Переходим в вершину 4: $s_0 = 4$. Вершина 4 стала изолированной, поэтому переходим к пункту 4 (б).

4 (б) Самая левая вершина нечетной кратности $s_0 = 5$ — крайняя точка. Переходим к пункту 6.

6. s_0 — крайняя точка, поэтому выполняем пункт 6 (а).

6 (а) $(s_0, s_1) = 10 = (5, 3)$.

7. Переходим к пункту 5.

5. Добавляем ребро 10 в T. $T = (7, 8, 9, 10) = ((1, 2), (2, 3), (3, 4), (5, 3))$. Удаляем ребро 10 из G: $G = (\{1, 2, 3, 4, 5, 6\}, \{11, 12\})$. Переходим в вершину 3: $s_0 = 3$. Вершина 3 не стала изолированной, поэтому переходим к следующему пункту.

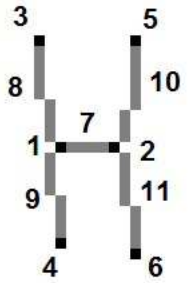

Рис. 5. Пример скелета рукописного символа H

6. $s_0 = 3$ — точка пересечения. Выполняем пункт 6 (б).

6 (б) В $U^2(3)$ первая пара — $(T_{|T|}, (s_0, s_1)) = (10, (s_0, s_1)) = (10, 11)$. При удалении ребра 11 связность графа G не нарушается.

7. Переходим к пункту 5.

5. Добавляем ребро 11 в T. $T = (7, 8, 9, 10, 11) = ((1, 2), (2, 3), (3, 4), (5, 3), (3, 2))$. Удаляем ребро 11 из G: $G = (\{1, 2, 3, 4, 5, 6\}, \{12\})$. Переходим в вершину 2: $s_0 = 2$. Вершина 2 не стала изолированной, поэтому переходим к следующему пункту.

6. $s_0 = 2$ — точка пересечения. Выполняем пункт 6 (б).

6 (б) В $U^2(2)$ первая пара — $(T_{|T|}, (s_0, s_1)) = (11, (s_0, s_1)) = (11, 12)$. При удалении ребра 12 связность графа G не нарушается.

7. Переходим к пункту 5.

5. Добавляем ребро 12 в T. $T = (7, 8, 9, 10, 11, 12) = ((1, 2), (2, 3), (3, 4), (5, 3), (3, 2), (2, 6))$. Удаляем ребро 12 из G: $G = (\{1, 2, 3, 4, 5, 6\}, \emptyset)$. Удалены все ребра графа, значит, поиск завершаем.

Найденный путь — $T = (7, 8, 9, 10, 11, 12) = ((1, 2), (2, 3), (3, 4), (5, 3), (3, 2), (2, 6))$.

§ 4. Пример графа с точками пересечения нечетной кратности

Рассмотрим пример скелета символа H (рис. 5). Выполним алгоритм поиска пути для графа этого символа. Исходный граф состоит из 6 вершин и 5 ребер: $G = (\{1, 2, 3, 4, 5, 6\}, \{(1, 2), (3, 1), (1, 4), (5, 2), (2, 6)\})$. Для удобства ребра обозначим через 7, 8, 9, 10, 11. Тогда граф примет вид $G = (\{1, 2, 3, 4, 5, 6\}, \{7, 8, 9, 10, 11\})$

1. Пусть $T = \emptyset$.

2. Граф G содержит 2 точки пересечения нечетной кратности: 1 и 2. Добавим 2 виртуальных ребра: $(1, 0) = 12$ и $(2, 0) = 13$. Тогда граф G примет вид $G = (\{1, 2, 3, 4, 5, 0\}, \{(1, 2), (3, 1), (1, 4), (5, 2), (2, 6), (1, 0), (2, 0)\}) = (\{1, 2, 3, 4, 5, 6, 0\}, \{7, 8, 9, 10, 11, 12, 13\})$.

3. В графе G всего 2 точки пересечения: 1 и 2.
$U^2(1) = (8, 9)(w = 1), (12, 7)(w = 0), (8, 7)(w = 2), (12, 9)(w = 0),$
$(9, 7)(w = 2), (12, 8)(w = 0);$
$U^2(2) = (10, 11)(w = 1), (7, 13)(w = 0), (10, 7)(w = 2), (11, 13)(w = 0),$
$(11, 7)(w = 2), (10, 13)(w = 0)$, где $w = 1$,$w = 2$,$w = 0$ — веса переходов.

4. Граф содержит вершины нечетной кратности, следовательно, выполняем пункт 4 (б).

4 (б) $s_0 = 3$ — крайняя точка. Переходим к пункту 6.

6. s_0 — крайняя точка, поэтому выполняем пункт 6 (а).

6 (а) $(s_0, s_1) = 8 = (3, 1)$.

7. Переходим к пункту 5.

5. Добавляем ребро 8 в T. $T = (8) = ((3, 1))$. Удаляем ребро 8 из G: $G = (\{1, 2, 3, 4, 5, 6, 0\}, \{7, 9, 10, 11, 12, 13\})$. Переходим в вершину 1: $s_0 = 1$. Вершина 1 не стала изолированной, поэтому переходим к следующему пункту.

6. $s_0 = 1$ — точка пересечения. Выполняем пункт 6 (б).

6 (б) В $U^2(1)$ первая пара — $(T_{|T|}, (s_0, s_1)) = (8, (s_0, s_1)) = (8, 9)$. При удалении ребра 9 связность графа G не нарушается.

7. Переходим к пункту 5.

5. Добавляем ребро 9 в T. $T = (8, 9) = ((3, 1), (1, 4))$. Удаляем ребро 9 из G: $G = (\{1, 2, 3, 4, 5, 6, 0\}, \{7, 10, 11, 12, 13\})$. Переходим в вершину 4: $s_0 = 4$. Вершина 4 стала изолированной, поэтому переходим к пункту 4 (б).

4 (б) Самая левая вершина нечетной кратности $s_0 = 1$ — точка пересечения, поэтому в $U^2(1)$ находим пару $((s_2, 1), (1, s_1))$, где $(s_2, 1)$ — виртуальное ребро. Виртуальным ребром является ребро 12, а искомой парой — $((s_2, 1), (1, s_1)) = (12, 7) = ((0, 1), (1, 2))$. Тогда $(s_0, s_1) = (1, 2) = 7$.

5. Добавляем ребро 7 в T. $T = (8, 9, 7) = ((3, 1), (1, 4), (1, 2))$. Удаляем ребро 7 из G: $G = (\{1, 2, 3, 4, 5, 6, 0\}, \{10, 11, 12, 13\})$. Переходим в вершину 2: $s_0 = 2$. Вершина 2 не стала изолированной, поэтому переходим к следующему пункту.

6. $s_0 = 2$ — точка пересечения. Выполняем пункт 6 (б).

6 (б) В $U^2(2)$ первая пара $(T_{|T|}, (s_0, s_1)) = (7, (s_0, s_1)) = (7, 13)$. При удалении ребра 13 связность графа G не нарушается. Ребро 13 виртуальное, переходим к пункту 4.

4. Граф содержит вершины нечетной кратности; следовательно, выполняем пункт 4 (б).

4 (б) $s_0 = 5$ — крайняя точка. Переходим к пункту 6.

6. s_0 — крайняя точка, поэтому выполняем пункт а).

6 (а) $(s_0, s_1) = 10 = (5, 2)$.

7. Переходим к пункту 5.

5. Добавляем ребро 10 в T. $T = (8, 9, 7, 10) = ((3, 1), (1, 4), (1, 2), (5, 2))$. Удаляем ребро 10 из G: $G = (\{1, 2, 3, 4, 5, 6, 0\}, \{11, 12, 13\})$. Переходим в вершину 2: $s_0 = 2$. Вершина 2 не стала изолированной, поэтому переходим к следующему пункту.

6. $s_0 = 2$ — точка пересечения. Выполняем пункт 6 (б).

6 (б) В $U^2(2)$ первая пара — $(T_{|T|}, (s_0, s_1)) = (10, (s_0, s_1)) = (10, 11)$. При удалении ребра 11 связность графа G не нарушается.

7. Переходим к пункту 5.

5. Добавляем ребро 11 в T. $T = (8, 9, 7, 10, 11) = ((3, 1), (1, 4), (1, 2), (5, 2), (2, 6))$. Удаляем ребро 11 из G: $G = (\{1, 2, 3, 4, 5, 6, 0\}, \{12, 13\})$. Удалены все ребра графа, следовательно, поиск завершаем.

Найденный путь — $T = (8, 9, 7, 10, 11) = ((3, 1), (1, 4), (1, 2), (5, 2), (2, 6))$.

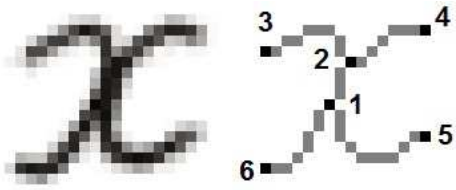

Рис. 6. Пример рукописного символа x

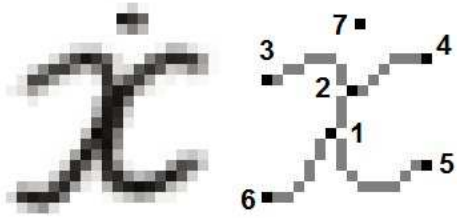

Рис. 7. Пример рукописного символа x с точкой

§ 5. Пример с двойной записью и изолированной точкой

Рассмотрим пример рукописного символа x и его скелета (рис. 6).

Граф содержит 5 ребер и 6 вершин нечетной кратности: $G = (\{1,2,3,4,5,6\}, \{(1,2),(2,3),(2,4),(1,5),(1,6)\})$. Из этого следует, что в последовательности записи должно быть как минимум 2 разрыва. После использования алгоритма поиска пути в зависимости от весов переходов может быть получен следующий результат: $T = ((3,2),(2,1),(1,6),(4,2),(1,5))$. Очевидно, что последовательность записи T не является оптимальным вариантом, и сложно в алгоритме распознавания предусмотреть такой вариант записи символа. Попробуем уменьшить количество разрывов в найденном пути. Очевидно, что для этого необходимо уменьшить количество вершин нечетной кратности. Добавим в граф G еще одно ребро $(1,2)$. Тогда граф выглядит следующим образом: $G = (\{1,2,3,4,5,6\}, \{(1,2)_1,(1,2)_2,(2,3),(2,4),(1,5),(1,6)\})$. Алгоритм поиска в этом случае дает следующий результат: $T = ((3,2),(2,1),(1,6),(4,2),(2,1),(1,5))$. Таким образом, разрыв в пути остался единственным, тем самым выполнено условие минимизации их количества.

Следует отметить, что добавление дополнительных ребер не всегда допускается. В данном случае из рис. 6 слева видно, что толщина кривой, соответствующей ребру $(1,2)$, больше, чем толщина остальных кривых. Из этого следует, что кривая $(1,2)$ записывалась не в один проход. Для поиска таких кривых может быть использована информация о толщине линий в пределах как отдельных компонент связности, так и всего текста в целом.

Если исходный граф содержит изолированные точки, то в пути, кроме ребер с указанием направления, добавляются эти изолированные вершины. Очевидно, что вершины добавляются в местах разрывов во избежание образования новых разрывов. Чаще всего точки выставляются после записи основной части символов. Для примера из рис. 7 это будет путь $T = ((3,2),(2,1),(1,6),(4,2),(2,1),(1,5),7)$.

§ 6. Заключение

Пусть t — некоторая последовательность координат на плоскости, задающая движение пишущего устройства, $B(t) = s$ — результат работы алгоритма online-распознавания [12–15], где s — результат распознавания (т. е. s — это символ, который получен в результате обработки последовательности t). Пусть T — путь в графе, найденный в результате применения алгоритма поиска пути с наименьшим весом. Так как путей с наименьшим весом может быть сразу

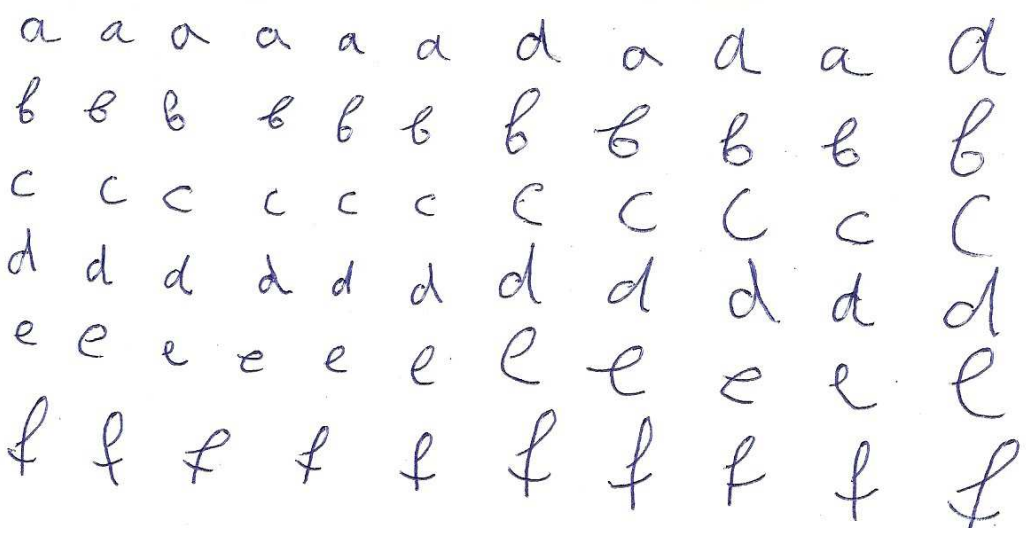

Рис. 8. Фрагмент тестовых данных

несколько, то $T = \{T_1, T_2, \ldots, T_n\}$. Пути тестовых символов будем обозначать через $T(s)$, где s — значение тестового символа. Будем считать, что траектория записи символа восстановлена правильно, если $\exists t \in T(s) : B(t) = s$. Иными словами, траектория записи восстановлена правильно, если среди путей с наименьшим весом найдется такой, который в результате обработки алгоритмом online-распознавания дает правильный результат.

Предложенный алгоритм был проверен на изображениях отдельных строчных рукописных латинских букв и цифр. Тестовые данные были сформированы таким образом, чтобы содержали варианты записи символов с разными размерами, наклонами и соотношениями длин компонент. Для каждого символа подготовлено по 20 вариантов записи. Фрагмент тестовых изображений приведен на рис. 8. Во всех 20 вариантах записи символов ‘c’, ‘e’, ‘i’, ‘j’, ‘l’, ‘m’, ‘n’, ‘o’, ‘u’, ‘v’, ‘w’, ‘x’, ‘z’, ‘0’, ‘1’, ‘2’, ‘3’, ‘6’, ‘9’ траектория записи восстановлена правильно. В символах ‘a’, ‘b’, ‘d’, ‘f’, ‘g’, ‘h’, ‘k’, ‘p’, ‘q’, ‘r’, ‘y’, ‘4’, ‘7’ правильно восстановлена траектория в 15–19 случаях из 20. Чаще всего ошибки возникают на символах ‘5’, ‘8’, ‘s’, ‘t’. В цифрах ‘5’, ‘8’ ошибка связана с тем, что запись должна начинаться не в критических точках, а в промежуточных. Для решения проблемы требуется доработать алгоритм, чтобы точки перегиба тоже рассматривались как вершины графа. В буквах ‘s’ и ‘t’ найденный путь не является наиболее оптимальным. Для решения проблемы требуется добавлять в базу дополнительные способы записи символов либо рассматривать пути не только с наименьшими весами, но и с близкими к наименьшим.

При обработке целых слов успех в большей степени зависит от выбора начальной точки. Большинство ошибок связано с тем, что по некоторым кривым запись должна выполняться сразу несколько раз. Предложенный алгоритм учитывает толщину линий, но для более точного анализа числа прохождений пишущего устройства по отдельным участкам требуются дополнительные исследования (например, учитывать интенсивность цветов линий).

Основным достоинством предложенного алгоритма над алгоритмами, использующими нейронные сети, является то, что не требуется сложного обучения, и механизм принятия решений интуитивно понятен. В отличие от других методов распознавания, которые используют структурный анализ, в данном случае не требуется выполнять посимвольную сегментацию. Стоит также отметить, что на точность работы алгоритма не сильно влияют случайные слияния линий и разрывы, так как важна не структура графа символа, а именно последовательность прохода по ребрам этого графа, которая выражается последовательностью координат на плоскости.

Предложенный алгоритм рассматривается не как готовый инструмент для распознавания

слитных рукописных текстов, а только как дополнение к online-алгоритмам. Также он может быть использован в комплексе с другими методами для увеличения точности распознавания.

СПИСОК ЛИТЕРАТУРЫ

1. Абраменко А. Принципы распознавания. К.: Компьютер-пресс, 1997. 123 с.
2. Кучуганов А.В., Лапинская Г.В. Распознавание рукописных текстов // Современные информационные технологии и письменное наследие: от древних рукописей к электронным текстам: материалы международной научной конференции (Ижевск, 13–17 июля 2006 г.). Ижевск: Изд-во ИжГТУ, 2006. С. 98–103. http://mns.udsu.ru/conf/report/Kuchuganov2.pdf
3. Омар М., Омар Ф., Исмоилов М.И., Остроух А.В. Применение систем распознавания образов в различных предметных областях // Автоматизация и управление в технических системах. 2014. № 4. С. 32–47. DOI: 10.12731/2306-1561-2014-4-4
4. Катасёв А.С., Катасёва Д.В., Кирпичников А.П. Распознавание рукописных символов на базе искусственной нейронной сети // Вестник Казанского технологического университета. 2015. Т. 18. № 11. С. 173–176. https://elibrary.ru/item.asp?id=23828611
5. Jaderberg M., Simonyan K., Vedaldi A., Zisserman A. Reading text in the wild with convolutional neural networks // International Journal of Computer Vision. 2015. Vol. 116. Issue 1. P. 1–20. DOI: 10.1007/s11263-015-0823-z
6. Фаворская М.Н., Горошкин А.Н. Модель распознавания изображений рукописного текста // Вестник Сибирского государственного аэрокосмического университета имени академика М. Ф. Решетнева. 2008. № 2 (19). С. 52–57. https://elibrary.ru/item.asp?id=25475486
7. Хаустов П.А. Алгоритмы распознавания рукописных символов на основе построения структурных моделей // Компьютерная оптика. 2017. Т. 41. № 1. С. 67–78. DOI: 10.18287/2412-6179-2017-41-1-67-78
8. Запрягаев С.А., Сорокин А.И. Распознавание рукописных символов на основе анализа дескрипторов функций длины хорды // Вестник Воронежского государственного университета. Сер. Системный анализ и информационные технологии. 2009. № 2. С. 49–58. https://elibrary.ru/item.asp?id=13040054
9. Спицын В.Г., Болотова Ю.А., Фан Н.Х., Буй Т.Т.Ч. Применение вейвлет-преобразования Хаара, метода главных компонент и нейронных сетей для оптического распознавания символов на изображениях в присутствии импульсного шума // Компьютерная оптика. 2016. Т. 40. № 2. С. 249–257. DOI: 10.18287/2412-6179-2016-40-2-249-257
10. Ray A., Rajeswar S., Chaudhury S. A hypothesize-and-verify framework for text recognition using deep recurrent neural networks // 2015 13th International Conference on Document Analysis and Recognition (ICDAR). Tunis, 2015. P. 936–940. DOI: 10.1109/ICDAR.2015.7333899
11. Ray A., Rajeswar S., Chaudhury S. OCR for bilingual documents using language modeling // 2015 13th International Conference on Document Analysis and Recognition (ICDAR). Tunis, 2015. P. 1256–1260. DOI: 10.1109/ICDAR.2015.7333965
12. Tapia E. Understanding mathematics: a system for the recognition of on-line handwritten mathematical expressions: Dissertation. Berlin, 2004. 109 p. https://pdfs.semanticscholar.org/2246/4fae30aa28706c323a0930ee82cb7f74856b.pdf
13. Kosmala A., Rigoll G. On-line handwritten formula recognition using statistical methods // Proceedings. Fourteenth International Conference on Pattern Recognition. Brisbane, Australia, 1998. P. 1306–1308. DOI: 10.1109/ICPR.1998.711941
14. Kosmala A., Rigoll G., Brakensiek A. Online handwritten formula recognition with integrated correction recognition and execution // Proceedings 15th International Conference on Pattern Recognition. Vol. 2. Pattern Recognition and Neural Networks. Barcelona, Spain, 2000. P. 590–593. DOI: 10.1109/ICPR.2000.906143
15. Toyozumi K., Suzuki T., Mori K., Suenaga Y. A system for real-time recognition of handwritten mathematical formulas // Proceedings of Sixth International Conference on Document Analysis and Recognition. Seattle, WA, USA, 2001. P. 1059–1063. DOI: 10.1109/ICDAR.2001.953948
16. Гонсалес Р., Вудс Р. Цифровая обработка изображений. М.: Техносфера, 2012. 1104 с.
17. Кристофидес Н. Теория графов. Алгоритмический подход. М.: Мир, 1978. 429 с.
18. Сапаров А.Ю., Бельтюков А.П. Математическое моделирование изображений формул с целью их распознавания // Вестник Удмуртского университета. Математика. Механика. Компьютерные науки. 2013. Вып. 1. С. 153–167. DOI: 10.20537/vm130114

19. Князев А.В. Распознавание слитных рукописных слов с помощью нечетких глирафов // Вестник МЭИ. 2017. № 5. С. 117–120. https://elibrary.ru/item.asp?id=30457194
20. Исупов Н.С., Кучуганов А.В. Использование теории графов в задаче распознавания рукописных текстов // Вестник ИЖГТУ им. М. Т. Калашникова. 2012. № 4. С. 160–162. https://elibrary.ru/item.asp?id=18973629
21. Сапаров А.Ю. Восстановление последовательности записи рукописных текстов в растровых изображениях // Десятая Всероссийская мультиконференция по проблемам управления МКПУ-2017: материалы 10-й Всероссийской мультиконференции (с. Дивноморское, Геленджик, Россия, 11–16 сентября 2017 г.). В 3-х томах. Т. 1. Ростов-на-Дону: Изд-во Южного федерального университета, 2017. С. 93–95. https://elibrary.ru/item.asp?id=29913561

Поступила в редакцию 06.07.2018

Сапаров Алексей Юрьевич, к. т. н., доцент, кафедра теоретических основ информатики, Удмуртский государственный университет, 426034, Россия, г. Ижевск, ул. Университетская, 1.
E-mail: say.saplh@gmail.com

A. Yu. Saparov

Recovering the recording sequence in scanned handwritten texts

Citation: *Vestnik Udmurtskogo Universiteta. Matematika. Mekhanika. Komp'yuternye Nauki*, 2018, vol. 28, issue 4, pp. 595–610 (in Russian).

Keywords: graph of a handwritten symbol skeleton, path in the skeleton, virtual edge.

MSC2010: 05C20, 68R10

DOI: 10.20537/vm180411

The article deals with the problem of recognizing handwritten texts from raster images. A method to recover the sequence of records in a handwritten text is described, that will reduce the task of offline-recognition to the task of online-recognition. The method is based on finding the Eulerian path with the minimum weight in the handwritten symbol skeleton graph. Some numerical characteristics are considered as weights, they show the complexity of the transition from one edge to another through a common vertex. A table of all possible combinations of pairs is constructed for this purpose. If there isn't Eulerian path in the original graph, the path is searched with the minimum number of breaks. The definition of a virtual edge is introduced, the transition on it is the formation of a gap in the path. It is necessary to split edges into pairs and calculate the weights at the vertices of odd multiplicity. The pathfinding algorithm in the skeleton of a symbol is considered, it is based on the Fleury's algorithm of searching Eulerian path.

REFERENCES

1. Abramenko A. *Printsipy raspoznavaniya* (Principles of recognition), K.: Computer-press, 1997, 123 p.
2. Kuchuganov A.V., Lapinskaya G.V. Recognition of handwritten texts, *Sovremennye informatsionnye tekhnologii i pis'mennoe nasledie: ot drevnikh rukopisei k elektronnym tekstam: materialy mezhdunarodnoi nauchnoi konferentsii* (Modern information technologies and written heritage: from ancient manuscripts to electronic texts: Proceedings of the International Scientific Conference), Izhevsk: Izhevsk State Technical University, 2006, pp. 98–103 (in Russian). http://mns.udsu.ru/conf/report/Kuchuganov2.pdf
3. Omar M., Omar F., Ismoilov M.I., Ostroukh A.V. Using of pattern recognition in various specialization, *Automation and Control in Technical Systems*, 2014, no. 4, pp. 32–47 (in Russian). DOI: 10.12731/2306-1561-2014-4-4
4. Katasev A.S., Kataseva D.V., Kirpichnikov A.P. Handwritten character recognition based on artificial neural network, *Vestnik Kazanskogo Tekhnologicheskogo Universiteta*, 2015, vol. 18, no. 11, pp. 173–176 (in Russian). https://elibrary.ru/item.asp?id=23828611

5. Jaderberg M., Simonyan K., Vedaldi A., Zisserman A. Reading text in the wild with convolutional neural networks, *International Journal of Computer Vision*, 2015, vol. 116, issue 1, pp. 1–20. DOI: 10.1007/s11263-015-0823-z
6. Favorskaya M.N., Goroshkin A.N. The invariant model for image recognition of hand-written text, *Vestnik Sibirskogo Gosudarstvennogo Aerokosmicheskogo Universiteta Imeni Akademika M. F. Reshetneva (Vestnik SibGAU)*, 2008, no. 2 (19), pp. 52–57 (in Russian). https://elibrary.ru/item.asp?id=25475486
7. Khaustov P.A. Algorithms for handwritten character recognition based on constructing structural models, *Computer Optics*, 2017, vol. 41, no. 1, pp. 67–78 (in Russian). DOI: 10.18287/2412-6179-2017-41-1-67-78
8. Zapryagaev S.A., Sorokin A.I. Handwritten character recognition based on analysis of chord length function descriptors, *Vestnik Voronezhskogo Gosudarstvennogo Universiteta. Ser. Sistemnyi Analis i Informatsionnye Tekhnologii*, 2009, no. 2, pp. 49–58 (in Russian). https://elibrary.ru/item.asp?id=13040054
9. Spitsyn V.G., Bolotova Yu.A., Phan N.H., Bui T.T.T. Using a Haar wavelet transform, principal component analysis and neural networks for OCR in the presence of impulse noise, *Computer Optics*, 2016, vol. 40, no. 2, pp. 249–257 (in Russian). DOI: 10.18287/2412-6179-2016-40-2-249-257
10. Ray A., Rajeswar S., Chaudhury S. A hypothesize-and-verify framework for text recognition using deep recurrent neural networks, *2015 13th International Conference on Document Analysis and Recognition (ICDAR)*, Tunis, 2015, pp. 936–940. DOI: 10.1109/ICDAR.2015.7333899
11. Ray A., Rajeswar S., Chaudhury S. OCR for bilingual documents using language modeling, *2015 13th International Conference on Document Analysis and Recognition (ICDAR)*, Tunis, 2015, pp. 1256–1260. DOI: 10.1109/ICDAR.2015.7333965
12. Tapia E. *Understanding mathematics: a system for the recognition of on-line handwritten mathematical expressions: Dissertation*, Berlin, 2004, 109 p. https://pdfs.semanticscholar.org/2246/4fae30aa28706c323a0930ee82cb7f74856b.pdf
13. Kosmala A., Rigoll G. On-line handwritten formula recognition using statistical methods, *Proceedings. Fourteenth International Conference on Pattern Recognition*, Brisbane, Australia, 1998, pp. 1306–1308. DOI: 10.1109/ICPR.1998.711941
14. Kosmala A., Rigoll G., Brakensiek A. Online handwritten formula recognition with integrated correction recognition and execution, *Proceedings 15th International Conference on Pattern Recognition. Vol. 2. Pattern Recognition and Neural Networks*, Barcelona, Spain, 2000, pp. 590–593. DOI: 10.1109/ICPR.2000.906143
15. Toyozumi K., Suzuki T., Mori K., Suenaga Y. A system for real-time recognition of handwritten mathematical formulas, *Proceedings of Sixth International Conference on Document Analysis and Recognition*, Seattle, WA, USA, 2001, pp. 1059–1063. DOI: 10.1109/ICDAR.2001.953948
16. Gonzalez R.C., Woods R.E. *Digital image processing*, Prentice-Hall, 2007, 976 p.
17. Christofides N. *Graph theory. An algorithmic approach*, Academic Press, 1975, 400 p.
18. Saparov A.Yu., Bel'tyukov A.P. Mathematical modeling of formula images for their recognition, *Vestnik Udmurtskogo Universiteta. Matematika. Mekhanika. Komp'yuternye Nauki*, 2013, issue 1, pp. 153–167 (in Russian). DOI: 10.20537/vm130114
19. Knyazev A.V. Recognition of joined-up handwritten words by means of fuzzy glyraphs, *Vestnik MEI*, 2017, no. 5, pp. 117–120 (in Russian). https://elibrary.ru/item.asp?id=30457194
20. Isupov N.S., Kuchuganov A.V. Graph theory application in handwriting recognition task, *Vestnik Izhevskogo Gosudarstvennogo Tekhnicheskogo Universiteta Imeni M. T. Kalashnikova*, 2012, no. 4, pp. 160–162 (in Russian). https://elibrary.ru/item.asp?id=18973629
21. Saparov A.Yu. Recovering the recording sequence in scanned handwritten texts, *10-ya Vserossiiskaya mul'tikonferentsiya po problemam upravleniya: Materialy mul'tikonferentsii* (The 10th All-Russian Multiconference on Management Problems: Multiconference materials), South Federal University, Rostov-on-don, 2017, vol. 1, pp. 93–95 (in Russian). https://elibrary.ru/item.asp?id=29913561

Received 06.07.2018

Saparov Aleksei Yur'evich, Candidate of Engineering, Associate Professor, Department of Theoretical Foundations of Computer Science, Udmurt State University, ul. Universitetskaya, 1, Izhevsk, 426034, Russia.
E-mail: say.saplh@gmail.com