ОБ ОДНОМ ПОДХОДЕ К АНАЛИЗУ ТОПОЛОГИИ ПРОСТРАНСТВЕННО-РАСПРЕДЕЛЕННЫХ ДАННЫХ С ИСПОЛЬЗОВАНИЕМ ЛОГИЧЕСКОГО ВЫВОДА

Р. К. Федоров, А. Е. Хмельнов, И. В. Бычков
Институт динамики систем и теории управления СО РАН, Иркутск, Россия
e-mail: bychkov@icc.ru

This paper addresses the problem of a road network recognition in a city. This problem belongs to a class of topologyrecognition problems on spatially distributed data. The initial data is represented indirectly, i. e, with help of objects located on the borders of the roads and the pedestrian footpasses. Heuristical algorithms and computer software for the recognition are constructed for both raster and vector cartographic data. During one of the main stages of the recognition process our method takes advantage of an automatic logical inference on the properties of the cartographic objects.

Введение

При решении практических задач геоинформатики (далее ГИС-задач), таких как подготовка и издание карт и атласов, оптимизация движения транспорта и др., пользователи сталкиваются с рядом проблем, усложняющих применение уже созданных векторных карт. Перечислим основные из них.

— Наличие в большинстве существующих векторных карт только *внутриобъектной топологии* [1], которая задает топологические отношения между примитивами (узлами и отрезками прямых) в пределах базовых составных элементов (полилиний, полигонов) картографических объектов[1]. Однако для решения многих ГИС-задач требуется *межобъектная топология*, которая описывает свойства картографических объектов в смысле их взаимного расположения, например, соседство и включение полигонов (многоугольников), принадлежность линии полигону и др. Выявление межобъектной топологической информации основывается на дополнительной информации — классификации объектов, как правило, не представленной на исходном картографическом материале. Заметим, что классификация картографических объектов зависит от решаемой в ГИС задачи. Примером задачи, использующей межобъектные топологические отношения, является задача

© Институт вычислительных технологий Сибирского отделения Российской академии наук, 2005.

[1]Под картографическим объектом в данной работе понимается картографическое представление любых элементов: воспроизводимых, хранимых, обрабатываемых на цифровых картах. Предполагается, что для каждого объекта сопоставляется некоторая семантика.

оптимизации движения транспорта. Ее решение невозможно без информации о связях между улицами.

— Наличие в исходных картографических данных большого количества топологических ошибок, возникающих, как правило, на стадиях векторизации картографического материала. Чаще всего это разрывы граничных линий, перехлесты площадных объектов и т. п. Для выявления и исправления топологических ошибок необходимо использовать информацию о классификации объектов.

— Разнообразие метрических представлений объекта на векторной карте. Объект можно представить несколькими способами: полигоном, полилинией, совокупностью различных других объектов, либо рассматриваемый объект может являться частью более сложного картографического объекта[2]. Часто основным критерием выбора представления объекта при векторизации картографического материала является визуальное сходство с бумажным вариантом. Для одной задачи более эффективно представление объекта в виде полилинии, для другой — в виде единого полигонального объекта (например, для подсчета площади). Практически невозможно создать векторную карту, которая бы эффективно представляла информационный ресурс для всей совокупности решаемых задач.

Решение этих проблем тесно связано между собой и относится к проблематике распознавания образов. Поэтому распознавание качественных характеристик (далее просто распознавание) картографических объектов является одной из основных задач обработки картографической информации.

В Институте динамики систем и теории управления СО РАН в рамках решения задачи распознавания дорожной сети населенного пункта разработан программный комплекс для анализа пространственно-распределенных данных. Этот комплекс обеспечивает построение модели дорожной сети как сложного картографического объекта на основе анализа межобъектной топологии векторного картографического материала. Задача усложняется тем, что в исходном картографическом материале дорожная сеть представлена в косвенном виде. На ней отсутствуют такие картографические объекты, как улица или перекресток, не говоря уже о связях между этими объектами. Информация об улицах представлена различными графическими примитивами, ограничивающими дорожное полотно (такими как бордюры), непосредственно примыкающими к проезжей части. Более того, нельзя сказать, с какой стороны от бордюра находится дорожное полотно.

Для решения этой задачи необходимо реализовать процедуру выделения области, принадлежащей дорожной сети. Такие области представлены граничными линиями, при этом данные картографические объекты не всегда обладают замкнутой граничной линией. Так, улицы могут иметь только боковые границы, в которых в свою очередь возможны разрывы. Поэтому для распознавания картографических объектов необходимо производить сегментацию, базируясь не только на граничных линиях. Следует производить декомпозицию области обработки на простые части (выделение графических примитивов).

На данный момент известны алгоритмы, которые могут эффективно выделить и классифицировать только простые формы, такие как круг, квадрат, эллипс и т. д. Очевидно, что в реальном мире и, как следствие, на векторной карте встречается континуум различных форм объектов. Картографические объекты, обладающие простой формой, встречаются довольно редко. Тем не менее, на карте населенного пункта такие объекты имеются в большом количестве. Этой формой обладают дороги, реки, тротуары и т. д. В дальнейшем в рамках этой статьи форму таких объектов будем называть линейной, а объекты

[2]Под сложным картографическим объектом далее будем понимать набор топологически связанных картографических объектов (частей, составляющих), имеющий/задающий определенную семантику.

линейными. Выделение линейных объектов может дать полезную информацию для последующего анализа городских территорий.

В реальных задачах на векторной карте большинство объектов представляют собой комбинации простых объектов (примитивов). Поэтому распознавание сложного объекта базируется на анализе свойств примитивов и их взаимного расположения. Очевидно, что сложные объекты обладают некоторой семантикой, на основе которой можно производить объединение как выделенных примитивов, так и сложных картографических объектов. Другими словами, распознавание сложного картографического объекта — это поиск набора топологически связанных картографических объектов, соответствующих его семантике. Семантика сложных картографических объектов может задаваться на основе семантики составляющих его объектов и их топологического расположения. Одним из способов задания семантики распознаваемых объектов является использование экспертных знаний в виде логических правил. Каждое правило задает либо набор объектов, каждый из которых обладает определенной семантикой и связан с другими определенным топологическим отношением, либо семантику некоторому объекту. Следовательно, распознавание сложных картографических объектов может базироваться на логическом выводе — применении внесенных в систему правил к заданному набору фактов, представляющему совокупность выделенных объектов и их признаков, в том числе топологические отношения между ними.

В программном комплексе решение этой задачи производится на основе известного подхода к анализу изображений [2], состоящего из трех этапов:

1) сегментация области обработки;
2) выделение признаков сегментов;
3) классификация сегментов на основе логических методов.

1. Сегментация области обработки

Обычно область исходного картографического материала (из нетопологических ГИС) не занята полигональными объектами полностью. Тем не менее, свободные области могут содержать необходимые картографические объекты или нести полезную для дальнейшей обработки информацию. Очевидно, что необходимо производить разбиение на полигональные объекты всей области обработки. Первоначально границами полигональных объектов (сегментов) будут являться граничные линии из исходных данных. Затем в соответствии с общим подходом будет производиться выделение из полученных полигональных объектов графических примитивов (линейных картографических объектов). Сегментация и выделение графических примитивов базируется на вспомогательных структурах — диаграмме Вороного и скелете диаграммы Вороного [3]. Диаграмма Вороного является средством, позволяющим упростить анализ метрики, так с ее помощью можно эффективно найти ближайшую точку или произвести трассирование контура объекта. Общая схема алгоритма сегментации области обработки состоит из следующих шагов:

— инициализация граничных точек;
— скелетизация области обработки;
— выделение линейных объектов;
— выделение вероятных разрывов.

Далее подробнее рассмотрим все шаги.

1.1. Инициализация граничных точек

В программном комплексе реализованы два способа ввода исходных данных:

— чтение и предобработка векторных файлов;

— векторизация растровых изображений с использованием модифицированного алгоритма нахождения границ Дж. Канни [4].

1.1.1. Чтение и предобработка векторных файлов

Исходные векторные данные считываются из одного или нескольких файлов, представленных в формате SHAPE. В большинстве векторных файлов границы объектов заданы упорядоченным набором отрезков, соединяющих точки. При вводе отрезки разбиваются на достаточно короткие фрагменты, при этом шаг разбиения отрезка выбирается из такого расчета, чтобы расстояние между соседними граничными точками было примерно вдвое меньше минимальной ширины распознаваемых объектов на карте.

1.1.2. Векторизация растровых изображений с использованием модифицированного алгоритма нахождения границ Дж. Канни

При обработке растровых изображений одной из основных задач является выделение границ между однородными областями изображения. Рассмотрим функцию яркости изображения $b = f(x, y)$, которая является непрерывной и дифференцируемой на множестве $X \times Y$. Если зафиксировать одну из переменных, например y, то график функции яркости на границе двух однородных областей X_j и X_k по переменной x можно представить в виде, как на рис. 1, где $x_1 \in X_j$ — точка, принадлежащая первой области, $x_3 \in X_k$ — точка, принадлежащая второй области, x_2 — граничная точка, разделяющая две области.

С учетом свойств представления растровых изображений далее рассматриваются дискретные значения яркости для функции $b = f(i, j)$, $i, j \subset N$ (рис. 2).

Для получения граничных точек реализован модифицированный алгоритм нахождения границ Дж. Канни. Границы объектов, найденные применением известного алгоритма Дж. Канни, имеют пиловидную форму из-за дискретного представления изображения, что снижает качество дальнейшего анализа изображений. Для получения границ, более соответствующих реальным границам объектов, предлагается учитывать практически подтвержденное свойство граничной точки: “совпадать с точкой перегиба функции изобра-

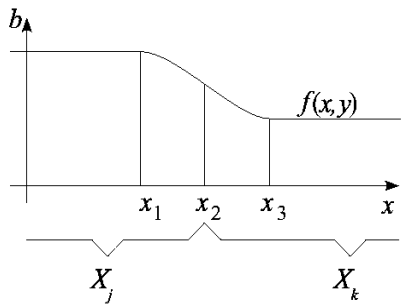


Рис. 1. Функция яркости изображения при фиксированной переменной y.

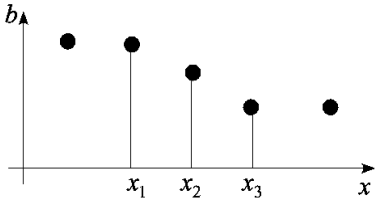


Рис. 2. Дискретизация функции яркости изображения.

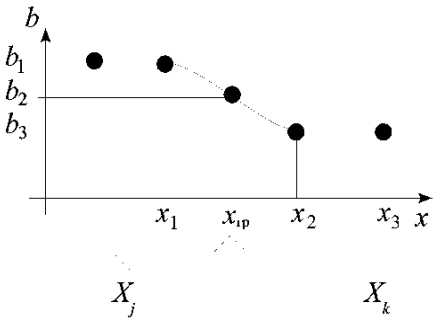


Рис. 3. График функции яркости изображения, $x_2 \in X_k$.

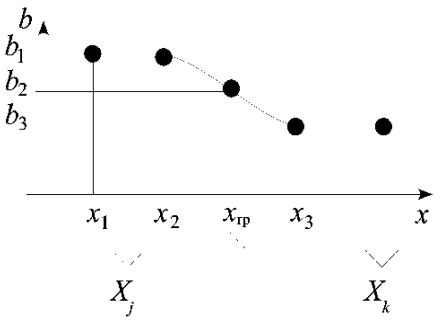


Рис. 4. График функции яркости изображения, $x_2 \in X_j$.

жения". Для нахождения точки перегиба строится аппроксимация функции яркости по дискретным значениям. Так как функция яркости является нелинейной, что требует для нахождения точки перегиба сложных вычислений, то для увеличения быстродействия алгоритма предложено следующее упрощение — использование линейной зависимости приращения граничной точки от значений яркостей x_1, x_2, x_3.

Рассмотрим два случая расположения граничной точки:

1. Граничная точка x_2 принадлежит области X_k. Тогда граница проходит посередине между точками x_1 и x_2 (рис. 3).

2. Граничная точка x_2 принадлежит области X_j. Тогда граница проходит посередине между точками x_2 и x_3 (рис. 4).

Положение приближенной границы можно выразить с помощью формулы: $x = x_1 + L(x_3 - x_1)$, где $L \in [0, 1]$.

Из рассмотренных случаев получаем, что при $b_2 = b_3$ $L = 0.75$, а при $b_2 = b_1$ $L = 0.25$.

Функция яркости серого непрерывна и дифференцируема, следовательно, уровень яркости граничной точки b_2 находится между двумя однородными областями с яркостями b_1, b_3, т. е. $b_2 \subset [b_1,\ b_3]$. На практике из-за присутствия шума значение b_2 может выйти за заданный интервал, чтобы не допустить этого, применяется сглаживающий фильтр Гаусса.

Таким образом, получаем следующее выражение для вычисления L:

$$L = \frac{0.5}{b_1 - b_3}(b_3 - b_2) + 0.75.$$

Выше был рассмотрен случай с фиксированным значением y. В двухмерном случае приращение граничной точки необходимо производить вдоль направления градиента изображения.

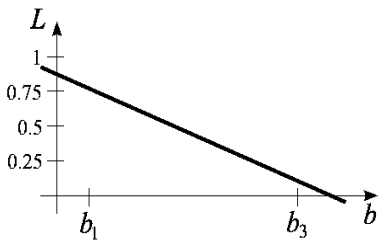
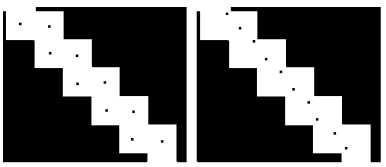


Рис. 5. График зависимости коэффициента L от яркости граничного пикселя.



а *б*

Рис. 6. Выделение границы. *а* — результат работы алгоритма Дж. Канни, *б* — результат работы модифицированного алгоритма.

Общая схема нахождения граничных точек на растровом изображении состоит из следующих шагов.

Шаг 1. Применение алгоритма Дж. Канни к исходному изображению и формирование набора граничных точек.

Шаг 2. Определение вектора сдвига каждой граничной точки. Для вычисления угла вектора сдвига рассматриваются точки, соседние для рассматриваемой граничной точки: сначала находится максимальная разница в яркости противоположных точек (относительно текущей), далее за угол вектора сдвига берется направление прямой, проходящей по противоположным точкам с максимальной разницей в яркости.

Шаг 3. Производится сдвиг граничных точек. Таким образом, предложенный алгоритм позволяет получить более гладкую линию границы (рис. 6).

1.1.3. Построение вспомогательных структур

После инициализации производится построение диаграммы Вороного на множестве граничных точек (рис. 7). Диаграмма Вороного — это разбиение плоскости на выпуклые многоугольники, которые называются ячейками диаграммы Вороного [3].

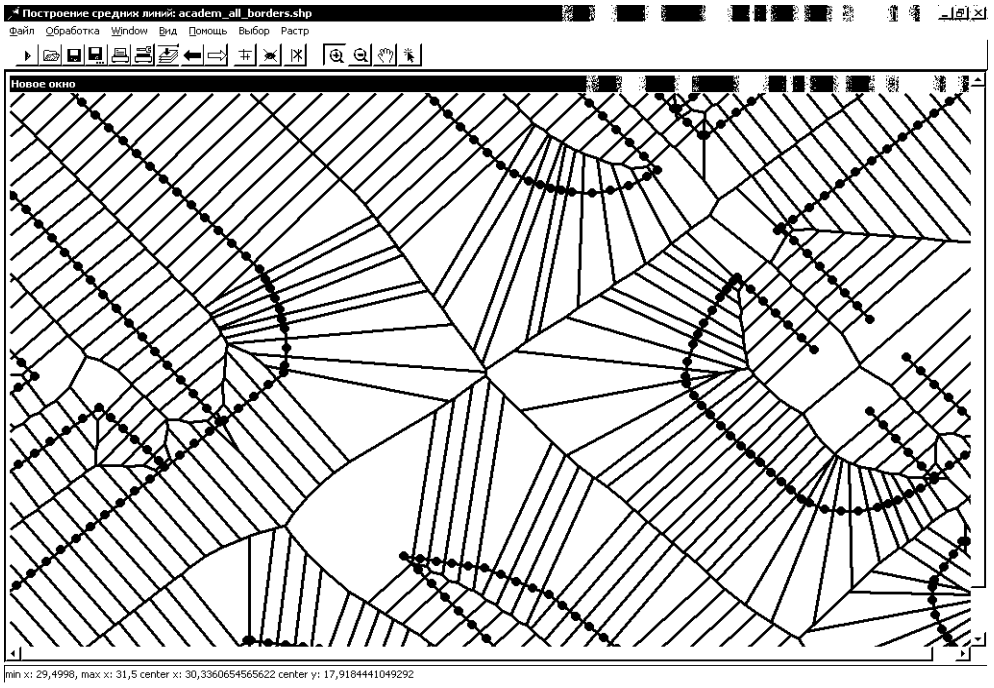


Рис. 7. Диаграмма Вороного, построенная на множестве граничных точек.

Ячейки диаграммы Вороного являются ограниченными многоугольниками, а для точек, лежащих в вершинах выпуклой оболочки диаграммы, границы между ячейками уходят на бесконечность. Отрезок или луч, разделяющий две соседние ячейки диаграммы, будем далее называть *ребром*, точку на конце такого отрезка — *вершиной*. Для представления лучей в число вершин включается специальная псевдоточка, обозначающая бесконечность.

Для построения диаграммы Вороного в комплексе используется алгоритм заметающей прямой (sweepline algorithm) [6], основанный на преобразовании Форчуна, который строит диаграмму за линейное по числу точек время.

После построения диаграммы Вороного на ней помечаются ребра, которыми разделяются соседние граничные точки. Эта информация используется в последующих алгоритмах. Далее эти ребра будем называть *граничными*. При этом делается предположение о том, что соседние граничные точки будут соседними и в диаграмме. Заметим, для того чтобы это предположение было верным и для векторных исходных данных, требуется шаг разбиения отрезков граничных линий объектов задавать достаточно малой величиной.

Для обнаружения граничных ребер были рассмотрены двойственные ребрам диаграммы Вороного ребра триангуляции Делоне, соединяющие точки, ячейки которых имеют общую границу, при этом по-разному обрабатываются изображения, представленные в векторном и растровом форматах. Так, на растровом изображении для проверки принадлежности ребра границе сравниваются цвета пикселов в его конечных вершинах, а в случае обработки информации, представленной в векторном формате, ребро считается граничным, если расстояние между разделяемыми им точками не превышает шаг разбиения. Такая проверка является максимально простой, но при этом, кроме точек, действительно находящихся рядом на одной ломаной, соединяются между собой и точки разных ломаных, если они расположены достаточно близко друг от друга.

В диаграмме Вороного необходимо выделить ребра, представляющие наибольший интерес. Такое множество важных ребер называется *скелетом* диаграммы Вороного (рис. 8).

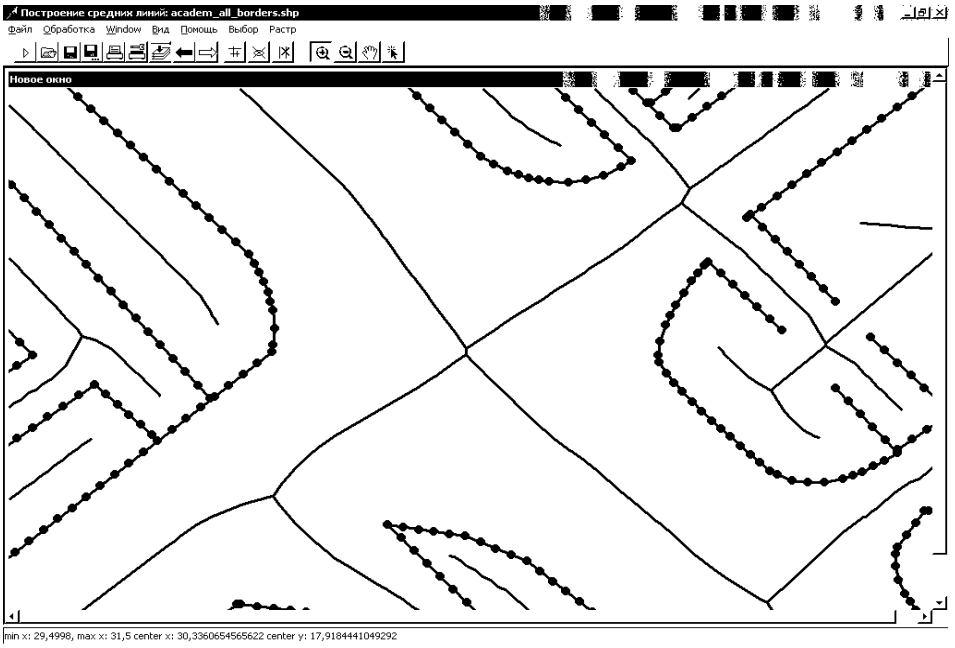


Рис. 8. Скелет диаграммы Вороного.

Для построения скелета диаграммы Вороного в соответствии со следующим правилами определяются значимые (скелетные) ребра:

1) отбрасываются граничные ребра, так как в основном эти ребра проходят между соседними точками границ объектов;

2) помечаются как скелетные:

— ребра, разделяющие различные объекты, т. е. если разделяемые ими точки относятся к разным связным компонентам графа граничных ребер;

— ребра, разделяющие граничные точки одного объекта, если расстояние между этими граничными точками по периметру достаточно велико. При определении расстояния между точками по периметру ищется кратчайший путь между точками на графе граничных точек объекта для рассматриваемой стороны этого объекта.

Диаграмма Вороного и ее скелет являются информационной основой, использующейся при дальнейшей обработке картографических данных программным комплексом.

1.2. Выделение скелетных линий линейных картографических объектов

Для выделения линейных объектов был разработан алгоритм, основывающийся на предположении, что все линейные объекты имеют некоторую ширину, которая изменяется либо достаточно медленно, либо скачкообразно. Рассмотрим некоторую функцию ширины объекта $f(l)$, зависящую от длины скелетной линии объекта l. Участки, соответствующие скелетной линии, будут линейными, если производная этой функции будет равняться нулю $f'(l) = 0$, на практике пользователь задает достаточно малую константу C, которая ограничивает производную $|f'(l)| < C$, для скелетной линии линейного объекта это выражение выполняется. При вычислении производной осуществляется аппроксимация зависимости ширины от длины скелетной линии, которая сглаживает дефекты представления данных.

Схема алгоритма выглядит следующим образом:

Шаг 1. Получение скелетного ребра.

Шаг 2. Трассирование скелетной линии и формирование двоек $(l, f(l))$.

Шаг 3. Аппроксимация полученных двоек совокупностью полиномов степени N методом наименьших квадратов. Для склейки полиномов применяется функция определенного вида:

$$f(x) = 1 - 3x^2 + 2x^3.$$

Шаг 4. Повторная трассировка ломаной и выделение участков, где значение модуля производной меньше заданной константы C.

Заметим, что недостатком этого алгоритма является неустойчивость к шумам, т. е. “лишние” граничные точки внутри линейного объекта могут повлиять на построение скелетной линии и на ее свойства.

1.3. Выделение вероятных разрывов

Шум, помехи на растровой карте, ошибки ввода векторных карт приводят к разрывам контуров объектов, т. е. для некоторых картографических объектов возможно отсутствие участка границы. Разрыв характеризуется наличием двух заканчивающихся линий, при этом если разрыв произошел локально (т. е. на относительно небольшом участке) на прямолинейном отрезке границы, то две заканчивающиеся линии имеют противоположные

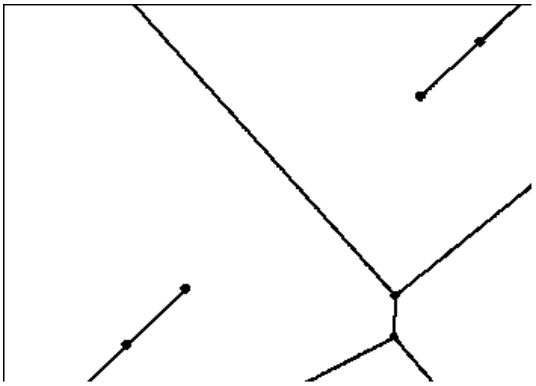


Рис. 9. Пример возможного разрыва контура объекта.

направления (рис. 9). На основании этого предположения в программном комплексе производят выделение вероятных разрывов. В векторных данных большинство разрывов являются локальными и чаще всего на прямолинейных участках границ. Такие разрывы возникают при векторизации картографического материала вследствие пересечения на растровом изображении границ картографических объектов, например, вследствие пересечения надписи с полигональным объектом. Алгоритм, находящий такие участки, работает при условии существования скелетного ребра, разделяющего конечные точки. На практике мы видим, что почти всегда такое ребро существует. Схема алгоритма выглядит следующим образом.

Шаг 1. Выбор скелетного ребра.

Шаг 2. Получение граничных точек, которые разделяет текущее скелетное ребро.

Шаг 3. Подсчет для каждой граничной точки количества соседних граничных точек, если обе граничные точки, разделяемые скелетным ребром, имеют по одному соседу, то эти две точки являются концами ломаных, иначе переход на шаг 1.

Шаг 4. Определение направления ломаных.

Шаг 5. Проверка, направлены ли эти ломаные друг к другу. Если они направлены, то помечаем текущее ребро как возможный разрыв.

Окончательное заключение о том, являются ли найденные вероятные разрывы результатом ошибок ввода картографических данных, производится на этапе классификации с использованием эвристик.

После выделения вероятных разрывов и линейных объектов закончено формирование сегментации области обработки. При этом границами объектов являются граничные линии и искусственно построенные отрезки. Алгоритм нахождения линейных объектов выделяет конечные скелетные ребра найденных линейных объектов. Алгоритм нахождения вероятных разрывов выделяет скелетные ребра, проходящие по ним. По парам граничных точек, разделенных выделенными скелетными ребрами, строятся искусственные отрезки.

2. Выделение признаков сегментов

В соответствии с парадигмой объектно-ориентированного программирования для каждого объекта создается экземпляр класса, в котором собраны реализованные алгоритмы выделения признаков сегментов. Класс имеет следующие операции и свойства:

`Nb` — массив указателей на объекты, имеющие общую граничную точку с данным объектом;

`Conts` — массив указателей на объекты, имеющие общее с данным объектом скелетное ребро;

`E` — указатель на скелетное ребро, находящееся внутри объекта;

`BndBox` — ограничивающая рамка;

`IsLine` — свойство, определяющее, является ли объект линейным;

`ForEachBndPoint` — итератор, обходящий по границе объекта;

`ConcatObject(A)` — операция присоединения объекта к данному;

`Draw` — операция отображения объекта.

Ниже приведены свойства, относящиеся только к линейным объектам:

`W` — ширина объекта;

`L` — длина объекта;

`Ang` — угол между осью объекта и осью $0x$.

Общую схему алгоритма инициализации экземпляров класса можно описать следующим образом:

Шаг 1. Получение текущего не обработанного скелетного ребра.

Шаг 2. Создание экземпляра класса.

Шаг 3. Перебор всех скелетных ребер в пределах текущего сегмента и установка их флагов как занятых.

Таким образом, для каждого сегмента (объекта) создается только один экземпляр класса. Данный объектно-ориентированный подход позволяет оперировать с набором экземпляров класса, а не с совокупностью граничных точек.

3. Классификация сегментов на основе логических методов

В основе алгоритма распознавания лежит использование логического вывода. Распознавание сложных картографических объектов — это сопоставление набора объектов, обладающих определенными свойствами и состоящих в определенных топологических отношениях друг с другом, набору правил (знаниям) интеллектной системы распознавания, реализованной в программном комплексе. Каждое правило задает либо сложный картографический объект как совокупность топологически связанных других объектов, либо семантику (свойство) объекта.

На начальном этапе обработки картографических данных производится сегментация области обработки с выделением линейных картографических объектов. Данные для логического вывода представляют собой набор площадных объектов (сегментов) и их признаков. Например, дорожная сеть состоит из линейных объектов и перекрестков. Однако не все линейные объекты являются улицами, среди них есть газоны, площадки и т. д. (рис. 10). Из-за имеющихся дефектов данных, например, таких как разрывы, невозможно определить простыми алгоритмами, к какому классу относится рассматриваемый линейный объект. Произвести классификацию объектов можно на основе таких признаков, как чередование дорог и газонов, связность дорожной сети и т. д.

При помощи правил представляются методики классификации объектов, кроме того, правила могут включать в себя различные эвристики, ориентированные на конкретные

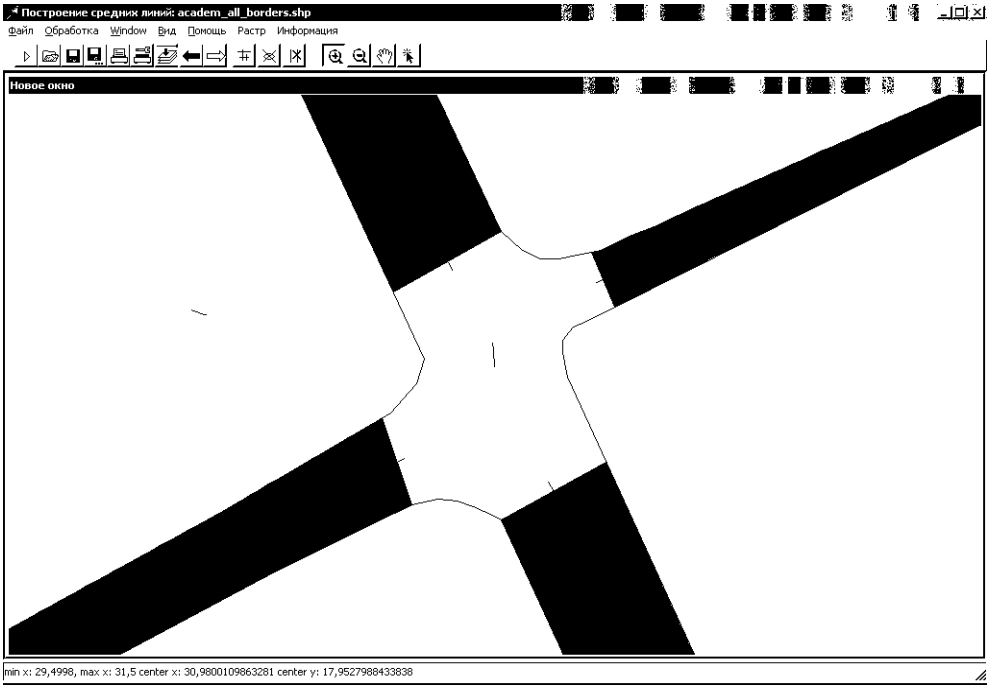


Рис. 10. Пример выделенных линейных объектов (фрагмент дорожного полотна с газонами и тротуарами).

данные. Набор правил распознавания дополняется пользователем, что позволяет ему настраивать работу программного комплекса на специфику своих данных и задачи. В качестве механизма логического вывода в программный комплекс встроен интерпретатор языка Пролог, соответственно, все правила распознавания представляются в виде правил этого языка.

Исходные данные предоставляются интерпретатору с помощью встроенных предикатов, истинность которых определяется по запросу в процессе логического вывода. Каждому предикату сопоставлена внутренняя функция, которая вычисляет признаки картографических объектов. Логический вывод производится над объектами, идентифицируемыми целыми числами.

3.1. Система встроенных предикатов

Ниже приведены встроенные предикаты и их описание:

`linearobject(V)` — истинный тогда и только тогда, когда объект `V` линейный;

`nblist(V,L)` — всегда истинный, возвращает список соседей, имеющих хотя бы одну общую граничную точку с данным объектом;

`contslist(V,L)` — всегда истинный, возвращает список соседей, имеющих хотя бы одно общее с данным объектом скелетное ребро;

`getobjflag(V,F)` — всегда истинный, возвращает тип объекта.

Для представления данных о линейных объектах определены следующие предикаты:

`against(A,B,C)` — истинный тогда и только тогда, когда объекты `A,C` находятся по разные стороны линейного объекта `B`;

`width(V,W)` — всегда истинный, возвращает среднюю ширину линейного объекта;

`length(V,L)` — всегда истинный, возвращает длину скелетной линии линейного объекта;

`getobjangle(V,F)` — всегда истинный, возвращает угол наклона линейного объекта;

`areonline(A,B)` — истинный тогда и только тогда, когда оба объекта лежат на одной линии.

Класс картографического объекта задается с помощью предиката `setflag(V,F)`, который всегда истинный.

Для получения класса объекта используется предикат `getobjtype(V,T)`.

Также в интерпретатор встроен предикат, выполняющий операцию объединения двух объектов, имеющих общее скелетное ребро:

`concat(V,V2)` — истинный тогда и только тогда, когда объединяются два объекта (к объекту `V` добавляется объект `V2` как его продолжение).

3.2. Логический вывод в процессе распознавания

Распознавание семантики сложного объекта — это логический вывод некоторой цели языка Пролог. Если конкретная цель выведена для заданного объекта, то считается, что объект обладает этой семантикой. То есть весь набор интересующих свойств объекта — это список выведенных для этого объекта целей. Вывод цели запускается для каждого объекта. Для получения объекта в Прологе встроен предикат `cur_obj(X)`, который возвращает текущий объект. Логический вывод производится до получения первого решения. Управляет запуском планировщик запуска целей, в нем записывается последовательность целей (сценарий), которые надо выполнить для получения конечного результата.

4. Применение программного комплекса

С помощью предложенного подхода были решены задачи представления карты дорожной сети на основе векторной карты г. Иркутск и создания дендрологической карты Академгородка. В статье мы рассмотрим решение задачи выделения дорожной сети.

Одной из востребованных ГИС-задач является оптимизация движения транспорта. Для решения рассматриваемой задачи дорожную сеть требуется представить в виде взвешенного графа дорог, где вершинами являются перекрестки и тупики, а дугами — осевые линии улиц. Однако приобретенная мэрией топографическая карта г. Иркутск не содержит данных о дорожной сети, поскольку улицы представлены такими объектами, как бордюры, площадки, дома и т. д., т. е. на карте отсутствуют объекты, представляющие улицы непосредственно. При этом информация об отдельном объекте, например бордюре, не позволяет сказать, с какой стороны от него находится дорожное полотно.

На основе входных данных в программном комплексе была произведена сегментация. Улицы соответствовали линейным объектам. Однако не все линейные объекты являлись улицами. Для извлечения улиц из набора линейных объектов реализована совокупность правил с использованием введенных предикатов, так, для определения улиц и газонов используются нижеследующие правила.

Правило для получения всех линейных объектов, примыкающих к возможному перекрестку:

```
getcrosslines([X|Cur],Prev,[X|Ans],C,M) :-
    C<M, linearobj(X),
```

```
    Cn is C + 1, getcrosslines(Cur,[X|Prev],Ans,Cn,M),!.
getcrosslines([X|Cur],Prev,Ans,C,M) :-
    C<M, arealobj(X), Cn is C + 1,
    getcrosslines(Cur,[X|Prev],Ans,Cn,M),!.
getcrosslines([X|Cur],Prev,Ans,C,M) :-
    C<M, contslist(X,NL),
    getabsences(NL,Sp,Prev), append(Cur,Sp,NCur), Cn is C + 1,
    getcrosslines(NCur,[X|Prev], Ans,Cn,M),!.
getcrosslines(_,_,[],C,M) :- C=M,!.
getcrosslines([],_,[],_,_).
```

Правило фильтрации линейных объектов, примыкающих к перекрестку. В соответствии с этим правилом все параллельные отрезки текущей линии, не являющиеся ее продолжением, отбрасываются:

```
fltr([X|S],[X|D],Beg) :-
    not(is_par(Beg,X)), !, fltr(S,D,Beg).
fltr([X|S],[X|D],Beg) :-
    is_par(Beg,X), areonline(Beg,X), !, fltr(S,D,Beg).
fltr([X|S],D,Beg) :- fltr(S,D,Beg).
fltr([],[],_).
```

Правило для определения улиц. Объект `X` является улицей, если он линейный и не имеет соседних параллельных линий:

```
is\_street(X) :-
    linearobj(X), not(thereisnbline(X)),
    setobjflag(X,1),!.
```

Объект `X` является улицей, если он соединяется не менее чем с двумя линейными объектами, примыкающими к перекрестку и не отсеченными правилом фильтрации, указанным выше:

```
is_street(X) :-
    linearobj(X), contslist(X,NL), member(Y,NL),
    getcrosslines([Y],[X],A,0,20), fltr(A,D,X), length(D,C1), C1>1,
    member(Z,NL), Z<>Y, not(is_deadend(Z,X,[])), setflags(D,1).
```

Правило определения перекрестков:

```
is_cross(X) :- not(arealobj(X)), contslist(X,NL), member(Y,NL),
    member(Z,NL), Z<>Y, isobjflag(Y,1), isobjflag(Z,1), setobjflag(X,1).
is_cross(X) :- not(arealobj(X)), contslist(X,NL), member(Y,NL), length(NL,C),
    C=1, isobjflag(Y,1), setobjflag(X,1).
```

Правило для определения тупиков:

```
is_deadend(X,Prev,His) :-
    arealobj(X), !, fail.
is_deadend(X,Prev,His) :-
    contslist(X,CNL), member(Y,CNL), arealobj(Y),!, fail.
```

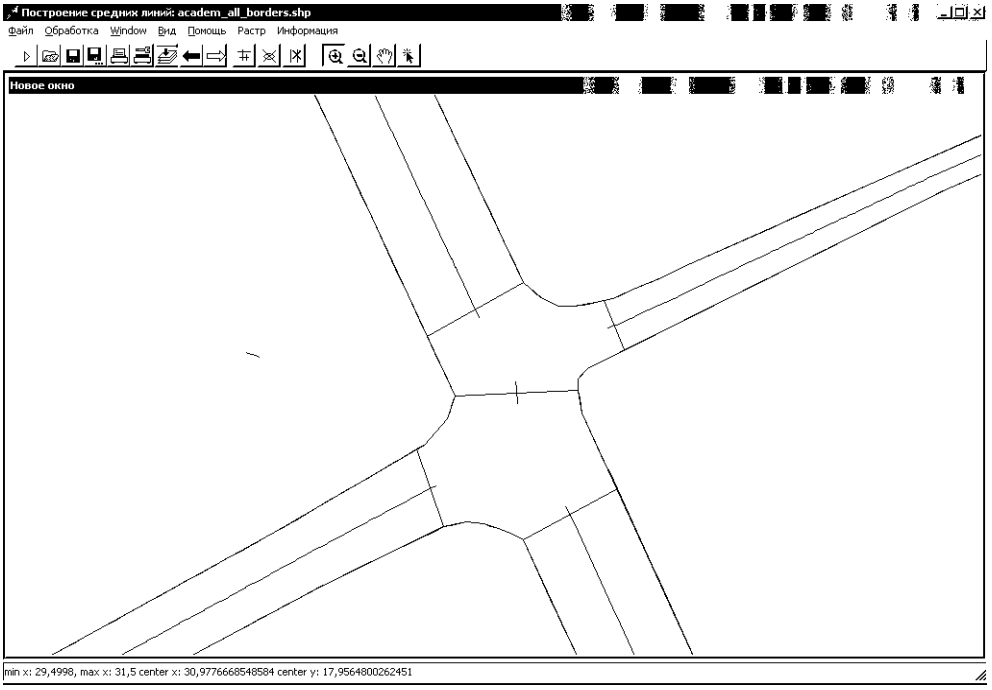


Рис. 11. Дорожная сеть.

```
is_deadend(X,Prev,_) :-
    contslist(X,CNL), length(CNL,C), C>2, !, fail.
is_deadend(X,Prev,His) :-
    contslist(X,CNL), length(CNL,C), C=2, !,
    member(Y,CNL), Y<>Prev, not(member(X,His)),
    is_deadend(Y,X,[X|His]).
is_deadend(X,Prev,_).
```

Ниже приведены примеры целей для выделения дорожной сети.

`?- curobj(X), is_street(X).` С помощью этой цели находятся все линейные дорожные объекты, примыкающие к перекресткам.

`?- curobj(X), isobjflag(X,1), contslist(X,NL), member(Y,NL), markonline(Y,X, X,0).` Эта цель предназначена для нахождения продолжений дорог, найденных предыдущей целью.

`?- curobj(X), is_cross(X).` Цель для определения перекрестков.

В результате построения модели картографических объектов и выполнения вывода целей получаем дорожную сеть (рис. 11).

Заключение

Предложенные алгоритмы позволили расширить класс решаемых задач, уменьшить количество ошибок распознавания, увеличить гибкость настройки программного комплекса. Программный комплекс реализован на языке Delphi v.5 и проходит этап опытной эксплуатации. В дальнейшем планируется расширить набор используемых предикатов и методов обработки картографических объектов.

Список литературы

[1] Горбачев В.Г. Что такое “топологические” отношения в цифровой картографии или для чего топологические отношения нужны в геоинформатике?
http://www.integro.ru/metod/topo_relations.htm

[2] Журавлев Ю.И., Никифоров В.В. Алгоритмы распознавания, основанные на вычислении оценок // Кибернетика. 1971. № 3. С. 1–11.

[3] Скворцов А.В. Триангуляция Делоне и ее применение. Томск: Изд-во Томск. ун-та, 2002. 128 с.

[4] Canny J. F. A computational approach to edge detection // IEEE Transactions on Pattern Analysis and Machine Intelligence. 1986. Vol. 8. P. 679–698.

[5] Fedorov R.K., Khmel'nov A.E. Road axial line builder // Pattern Recognition and Image Analysis. 2003. Vol. 13, N 2. P. 256–258.

[6] Fortune S.J. A sweepline algorithm for voronoi diagrams // Algorithmica. 1987. N 2. P. 153–174.

Поступила в редакцию 25 мая 2004 г.,
в переработанном виде — 30 ноября 2004 г.