

ЭФФЕКТИВНЫЕ АЛГОРИТМЫ ГЕОМЕТРИЧЕСКОГО ПОИСКА ЭЛЕМЕНТОВ ВЕКТОРНЫХ КАРТ

С. В. Куделич

В настоящее время значительно возросла область применения данных спутниковых наблюдений в таких областях как ГИС [1, 2], электронная картография [3, 4, 5], исследование природных ресурсов, мониторинг и оценка последствий стихийных бедствий и антропогенного воздействия на окружающую среду, планирование и управление развитием городской инфраструктуры, сельское и лесное хозяйство, туризм и др.

Системы электронной картографии позволяют оперативно получать информацию о местоположении, типе, количестве и параметрах объектов в любой точке Земного шара. Это упрощает процесс работы с геопространственной информацией, а так же открывает большие возможности для анализа, планирования и принятия решений на локальном и региональном уровнях.

Системы, использующие векторные карты, оперируют большим числом векторных объектов. Как правило, число векторных объектов настолько велико, что не представляется возможным загрузить их все в оперативную память. Более того, необходимо осуществлять быстрый поиск объектов, попадающих в определенную область.

Для преодоления данных проблем можно использовать реорганизацию исходного пространства данных, а также использовать динамическую многоуровневую подкачку данных с разной степенью детализации.

СТРУКТУРА МОДЕЛИ

Для эффективного поиска объектов необходимо реорганизовать исходное пространство данных. Для разбиения пространства необходимо использовать иерархические деревья (квадроструктуры) как наиболее эффективные структуры данных для пространственного поиска. Очевидно, что для поиска объектов нам необходимо реорганизовать только координаты вершин объектов, а дополнительные атрибуты объектов, такие как тип, цвет и т.д. можно хранить отдельно в динамической базе данных.

Наша модель для хранения, поиска и отображения объектов состоит из двух частей:

1. Бинарный файл, содержащий в себе иерархическое разбиение всей области (карты) с помощью оптимизированного 2d-дерева [6].

2. База данных SQLite [7], содержащая координаты вершин векторных объектов, а так же дополнительные атрибуты, такие как тип объекта, порог видимости объекта, цвет объекта.

Для быстрого поиска объектов, попадающих в целевую область отображения, в нашей модели используется оптимизированное 2d-дерево как наиболее быстрая древовидная структура для иерархического разбиения двумерных областей. Поскольку объекты задаются координатами вершин, то в дереве мы храним координаты вершин разбиения и идентификаторы всех объектов, хотя бы частично попадающих в ячейку.

Выбор базы данных SQLite обусловлен хорошими временными характеристиками скорости обработки запросов.

ПЕРЕМЕЩЕНИЕ ПО КАРТЕ И ПОДКАЧКА ДАННЫХ

Для быстрого отображения данных мы будем хранить части области карты вокруг области отображения на трех уровнях детализации (уровень «ниже», «текущий» уровень, уровень «выше»). Каждый уровень представляет собой 9 областей части карты при текущем масштабе. Это значит, что, если мы хотим отобразить на экран область карты размером $A \times B$, то на «текущем» уровне будут храниться 9 областей размером $A \times B$ (одна центральная и 8 обрамляющих), на уровне «выше» будут храниться 9 областей размером $A1 \times B1$, где $A1$ и $B1$ в два раза больше, чем A и B соответственно (т.е. имеем масштаб в два раза меньше) и на уровне «ниже» будут храниться 9 областей размером $A2 \times B2$, где $A2$ и $B2$ в два раза меньше, чем A и B соответственно (т.е. имеем масштаб в два раза больше). Каждая из 9 областей любого из трех уровней детализации будет представлена в памяти битовым изображением фиксированного размера (размер битового изображения для всех областей одинаковый) (рис.1). Это позволяет нам не хранить в памяти все векторные объекты, попадающие в некоторую область, а хранить только битовые изображения, на которых данные объекты были отрисованы с соответствующим масштабом.

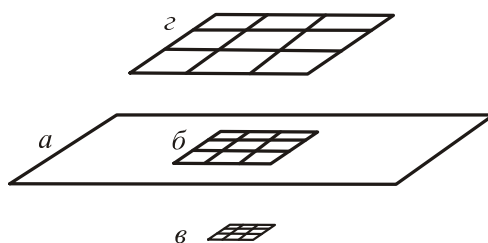


Рис. 1. Три уровня детализации:
а – карта, б – «текущий» уровень, в – уровень «ниже», г – уровень «выше»

Поскольку каждый уровень представляется в памяти девятью изображениями, то изображение на экране при движении на текущем уровне мы формируем из данных девяти частей. При этом большая часть изображения формируется из центральной области уровня.

Когда для отрисовки области отображения на «текущем» уровне необходимо более половины одной из обрамляющих областей вдоль любого измерения, необходимо произвести подкачку нового слоя и удалить освободившийся слой (рис. 2).

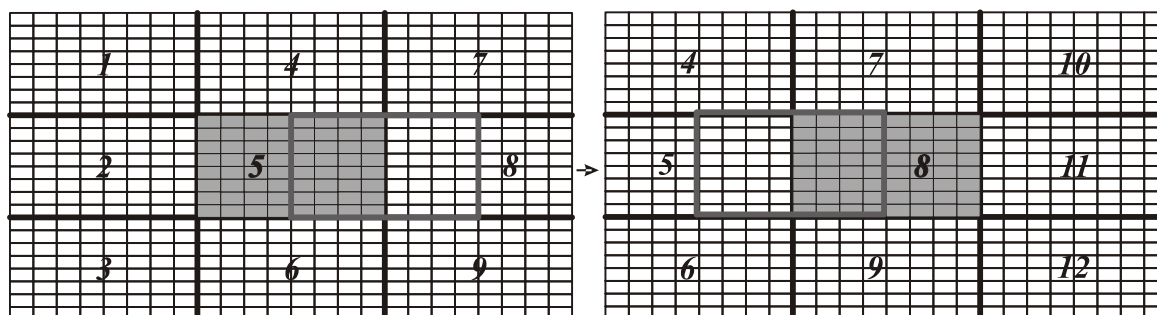


Рис. 2. Подкачка нового слоя при движении вправо:
области 1, 2, 3 освобождаются, области 10, 11, 12 подгружаются

При движении на «текущем» уровне нам также необходимо подгружать слои на уровне «выше» и на уровне «ниже». На уровне «выше» слои подгружаются в два раза реже, так как верхний уровень имеет в два раза больший размер карты, проецируемый в область отображения. На уровне «ниже» слои подгружаются в два раза чаще, так как этот уровень имеет в два раза меньший размер карты.

Движение между уровнями может происходить в двух направлениях:

1. На уровень «выше» – с увеличением размера отображаемой части карты.
2. На уровень «ниже» – с уменьшением размера отображаемой части карты.

Для реализации плавного и быстрого движения между уровнями мы перед переходом на уровень «ниже» «растягиваем» часть карты «текущего» уровня с текущей детализацией. А при переходе на уровень «выше» мы «сжимаем» часть карты «текущего» уровня. Целесообразно использовать несколько таких промежуточных уровней «сжатия» («растяжения»), это позволит выиграть время на загрузку нового уровня (рис. 3).

При переходе через последний уровень сжатия (растяжения) мы меняем «текущий» уровень на уровень «выше» («ниже»), используем «растяжение» («сжатие») на нем и производим подгрузку нового уровня.

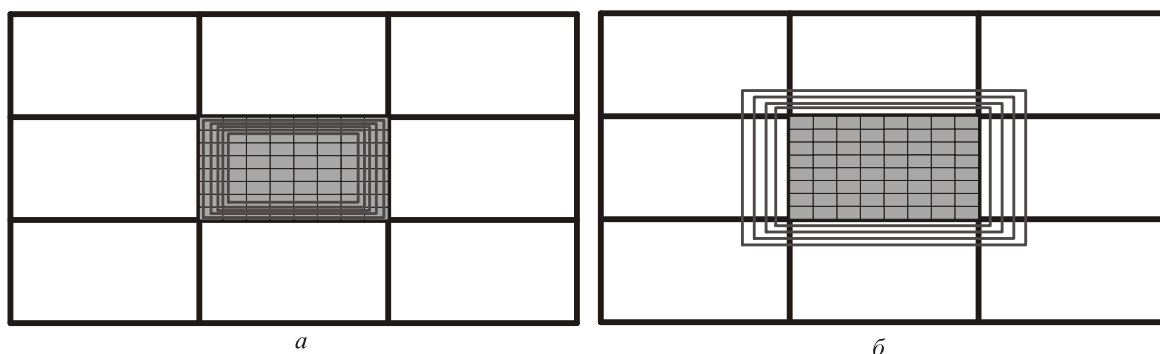


Рис.3. Растяжение и сжатие «текущего» уровня:
а –растяжение, б – сжатие

ПРЕИМУЩЕСТВА МЕТОДА

При работе с картографическими системами важное место отводится скорости поиска объектов и объему оперативной памяти, требуемой для работы приложения.

Главным преимуществом данного подхода является время поиска объектов, попадающих в целевую область. При использовании тривиального последовательного поиска среди N объектов время запроса составляет $O(N)$. При использовании оптимизированного 2-d дерева время запроса в среднем составляет $O(\log N + k)$, где k – число объектов, попавших в область. В современных картографических системах используются такие иерархические структуры данных как квадродеревья, матричные квадродеревья и оптимизированные квадродеревья [8]. Не смотря на то, что эти структуры данных имеют в среднем такую же асимптотику времени запроса как и оптимизированное 2-d дерево, на практике они в 2-5 раз уступают по скорости оптимизированному 2-d дереву.

Так же важным преимуществом данного подхода является использование битовых изображений фиксированного размера. Это позволяет использовать небольшой фиксированный объем оперативной памяти для просмотра векторных карт любого размера, что дает возможность запускать картографическую систему даже на устройствах с небольшим объемом оперативной памяти.

ЗАКЛЮЧЕНИЕ

Использование реорганизации исходного пространства данных с помощью оптимизированного 2-d дерева, а так же использование динамической многоуровневой подкачки данных с разной степенью детализации позволяет работать с векторными картами очень больших размеров и перемещаться по ним с достаточно хорошей скоростью без видимых задержек.

Литература

1. Варфоломеев И. В., Ермакова И. Г., Савельев А. С. Алгоритмы и структуры данных геоинформационных систем. Красноярск: КГТУ. 2003.
2. Ефремов В. Ю., Крашенинникова Ю. С., Лупян Е. А., Мазуров А. А., Прошин А. А., Флитман Е. В.. Оптимизированная система хранения и представления географически привязанных спутниковых данных: Сборник научных статей. М.: Полиграф сервис, 2004. С. 437–443.
3. Интернет-адрес: <http://maps.google.ru>.
4. Интернет-адрес: <http://maps.yandex.ru>.
5. Интернет-адрес: <http://www.google.com/earth/index.html>.
6. Bentley J. L. Multidimensional binary search trees used for associative searching// Communications of the ACM, Volume 18 Issue 9, Sept. 1975. P. 509–517.
7. Grant Allen, Mike Owens. The Definitive Guide to SQLite (Second Edition). 2010.
8. Препарата Ф. Вычислительная геометрия : введение. М.: Мир. 1989.

ТЕХНОЛОГИИ ПОИСКА ЛИЦ В БАЗЕ ДАННЫХ ИЗОБРАЖЕНИЙ

В. А. Куликович

ВВЕДЕНИЕ

Существует большое разнообразие методов идентификации личности, причем многие из них получили широкое коммерческое применение и используются уже немало лет. В последнее время активно развиваются методы биометрической идентификации, позволяющие определить личность человека по его физиологическим характеристикам путем распознавания по образцам. Одним из носителей таких характеристик является лицо.

Идентификация человека по его лицу находит применение в широком круге приложений: системы контроля удостоверений личности, криминалистика, информационная безопасность.

В работе предлагается подход к решению задачи поиска. Разработан алгоритм выделения геометрических характеристик лица, который выполняет сопоставление с базой изображений. В программной реализации разработанных методов используются возможности многофункциональной библиотеки компьютерного зрения OpenCV.

ПОСТАНОВКА ЗАДАЧИ

Целью работы является разработка и реализация последовательности алгоритмов поиска изображения лица по базе.