Эффективный алгоритм построения остова растрового изображения

Денис Иванов, Евгений Кузьмин
Механико-математический факультет
Московский Государственный Университет
Москва, Россия

Аннотация

Эта статья посвящена эффективному с точки зрения времени и памяти алгоритму построения остова (скелета) растрового изображения. Остов является линейным объектом, полностью характеризующим растровую область, и его построение позволяет решить задачу преобразования растрового изображения в векторное представление.

В данной работе подробно обсуждается понятие остова растровой области и даётся его формальное определение. Затем вводится понятие хорды для пиксельной строки и с его помощью строится быстрый алгоритм отыскания остова. В заключение пассмотрены различные возмовные применения полученного представления растра.

Ключевые слова: *растр, остов, векторизация.*

1. ВВЕДЕНИЕ

С развитием вычислительной базы значительно расширилась сфера применения систем автоматизированного проектирования и производства (САПР) и геоинформационных систем (ГИС). Так как большинство таких систем ориентировано на работу с векторной графикой, возникает задача о преобразовании уже имеющихся растровых документов, полученных с помощью сканера или цифровой камеры, в векторное представления[1]. Подобное преобразование является также основой для анализа в системах оптического распознавания символов (ОРС).

На сегодняшний день наиболее изучены два подхода к решению упомянутой проблемы[2]. Первый основан на многократной фильтрации исходного изображения, с посложным удалением граничных пикселей растровых областей. После многократного применения такого фильтра области редуцируются в линейные объекты, которые и являются основой векторного представления. Второй подход к решению поставленной задачи состоит в прослеживании линий, проходящих внутри области. При этом, начиная линию в некоторой точке, алгоритм каждый раз выбирает наилучшее направление смещения по некоторому критерию, и тем самым, непосредственно строит линию.

Следует заметить, что оба предлагаемых метода имеют следующие недостатки:

- Для анализа требуется хранение всего исходного растрового изображения (или, по крайней мере, большого его фрагмента) в памяти компьютера, что, с учётом обычных размеров в несколько мегабайт, является неэкономичным;
- Оба алгоритма требуют изучения свойств каждого текущего пикселя, что приводит к огромным временным затратам на преобразование.

В данной работе предлагается оригинальный подход к решению проблемы построения векторного представления растрового изображения, основанный на итеративном анализе строк растра. Описываемый алгоритм позволяет оптимально с точки зрения времени и памяти построить остов исходного растра. Рассматриваются различные возможности последующей обработки остова.

2. ОСТОВ РАСТРОВОЙ ОБЛАСТИ

До описания алгоритма отыскания остова растровой области, введем сначала его строгое, формальное определение. С этой целью будут рассмотрены специальные функция и поверхность, построенные над растровой плоскостью.

Условимся рассматривать 4-связные растровые области, то есть такие, что любые два её пикселя можно соединить 4-связной линией, лежащей внутри области.

2.1 Дискретное пространство на растре

Поместим растр в прямоугольную евклидову систему координат, так чтобы пиксели исходного изображения накрывали квадраты со стороной длины два. Сопоставив каждому квадрату атрибут – его цвет, получим представление исходного изображения. Рассмотрим дискретное подпространство данного пространства, содержащее все точки с целочисленным координатами (каждому пикселю будет соответствовать 9 точек). Это пространство будем называть дискретным пространством на растре и в дальнейшем, если не указано особо, все понятия будем определять для него.

2.2 Функция расстояния до границы

Рассматривая дискретное двумерное пространство точек на растровой плоскости, введём на ней метрику, обозначаемую

$$L_{\infty}^{2} : \rho(P,Q) = \max(|Px - Qx|, |Py - Qy|)$$

Для заданной 4-связной области можно теперь определить функцию расстояния до границы области

$$R(x,y) = \rho(P(x,y), \partial\Omega), \text{ где}$$

$P(x,y)$- точка растра с координатами (x,y), а

$\partial\Omega$ - граница данной области.

Полученная функция имеет следующие свойства:

- $R(x,y)$ неотрицательна;
- $R(x,y)= 0$ в граничных точках области.

В дальнейшем, нами будут рассматриваться лишь значения функции в точках, принадлежащих исходной растровой области.

2.3 Поверхность над областью

Рассмотрим график функции расстояния до границы данной пиксельной области $R(x,y)$, направив третью ось перпендикулярно растровой плоскости. Получим поверхность в трехмерном пространстве, построенную над двумерной областью. Примеры простейших поверхностей показаны на Рис.1.

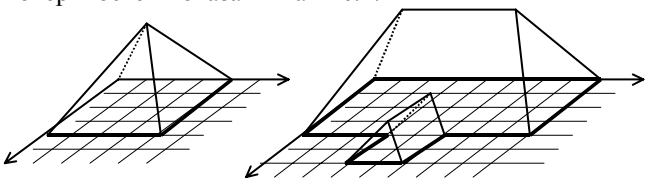

Рис.1. Пример поверхностей над пиксельной областью

Так как построенные поверхности соответствуют функциями расстояния до границы, то можно говорить об их “непрерывности” в том смысле, что значения функции $R(x,y)$ в двух соседних точках не могут отличаться более чем на единицу.

Нетрудно установить следующие свойства поверхности над растровой областью:

- Она не имеет точек локального минимума, за исключением точек границы области;
- Любой отрезок границы области определяет грань, принадлежащую данной поверхности;
- Не существует поверхности над областью, содержащей горизонтальные грани.

Перечисленные свойства дают представление о форме определённой поверхности, а именно, понятно, что она состоит из граней, хотя бы одна из сторон которых принадлежит границе области. Построенная таким образом поверхность является основополагающим понятием для определения остова.

2.4 Ребро

Рёбрами данной области будем называть отрезки пересечения граней её поверхности.

Вспомнив, что рассматриваемая поверхность есть график функции расстояния до границы $R(x,y)$, и установив по определению, что точки, принадлежащие ребру, принадлежат не менее чем двум граням, легко можно понять следующее утверждение:

Пусть точка (x,y,r) принадлежит ребру, тогда расстояние от точки (x,y), лежащей на растровой плоскости, до граничных точек области достигает своего минимума более чем в одной точке, другими словами, точка (x,y) равноудалена от не менее чем двух частей границы.

Этот факт даёт право считать понятие ребра важным для определения остова растровой области.

2.5 Скелет области

Обозначим $\{R_i\}_{i=1...n}$ - множество всех рёбер поверхности для заданной 4-связной пиксельной области. Скелетом этой области будем называть объединение всех этих рёбер, то есть

$$S = \bigcup_{i=1}^{n} R_i .$$

Пример проекции скелета простейшей области на растровую плоскость показан на Рис.2.

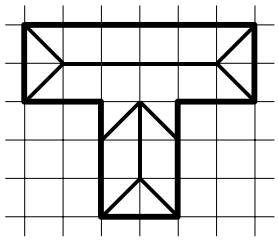

Рис.2. Пример скелета растровой области

Как уже замечалось, точки плоскости, соответствующие рёбрам равноудалены от не менее чем двух различных частей границы, и следовательно несут информацию о форме области. К тому же скелет пиксельной области имеет следующее важное свойство, обеспечивающее его уникальность.

2.5.1 Теорема об однозначности скелета

Теорема

Соответствие область-скелет является взаимно однозначным, то есть область однозначно представима своим скелетом.

Доказательство.

Докажем, что

$$\Omega = \bigcup_{(x,y,r)\in S} K((x,y),r) \text{ , где}$$

Ω - исходная область,

$K((x,y),r)$ - круг на растровой плоскости с центром в точке *(x,y)* и радиусом *r* (круг рассматривается в выбранной метрике), а

объединение берётся по всем точкам, принадлежащим скелету.

Каждый рассматриваемый круг по определению лежит внутри области (напомним, что его радиус есть расстояние до границы от центра), следовательно и их объединение так же лежит в области.

Осталось доказать, что каждая точка области накрывается хотя бы одним кругом. От противного, пусть точка (*x*,*y*) лежит в области, но не принадлежит никакому кругу. Рассмотрим соответствующую ей точку (*x*,*y*,*r*) на поверхности и грань этой поверхности, её содержащую. Восстановим перпендикуляр от точки (*x*,*y*,*r*) к части границы, порождающей данную грань, и продлим его в другую сторону до пересечения с соответствующим ребром. Полученную точку обозначим *P* (см. Рис.3).

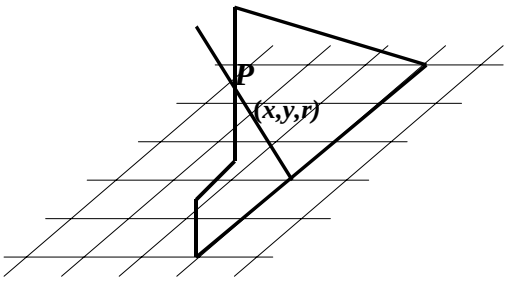

Рис.3. К доказательству теоремы об однозначности скелета

Очевидно, что точка *P* принадлежит скелету, так как лежит на ребре, и круг с центром в проекции этой точки на плоскость и соответствующим радиусом накрывает точку с координатами (*x*,*y*). Этот факт противоречит нашему предположению, и, следовательно, завершает доказательство теоремы.

Конец доказательства.

Хотя скелет и является однозначным линейным представлением исходной растровой области, он имеет два недостатка:

- *Избыток информации.* Назовём рёбра, одна концевая точка которых лежит на границе исходной области, угловыми (см. Рис.4).

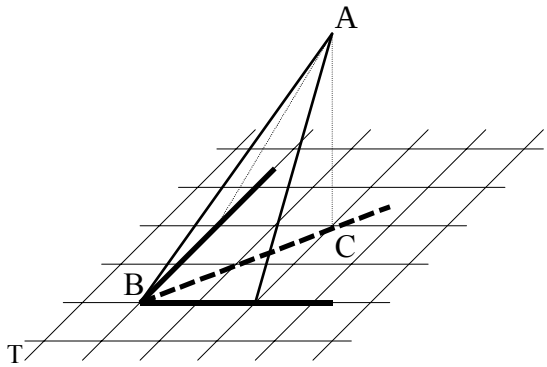

Рис.4. Пример углового ребра.

Круг с центром в точке *C* – проекции другой концевой точки на плоскость, и соответствующим радиусом полностью накроет отрезок [*BC*], проекцию данного ребра, а значит и все круги, восстановленные в промежуточных точках этого отрезка. Следовательно, для восстановления области достаточно иметь лишь одну концевую точку вместо угловых рёбер.

- *Несвязность скелета.* Вторым недостатком скелета можно считать его несвязность. Действительно, скелет не является связным множеством, как это видно из Рис.2.

Принимая во внимание упомянутые недостатки формально определённого скелета, а именно его избыточность и несвязность, определим наконец тот объект, который мы будем считать точным векторным представлением растровой области.

2.6 Остов

Введём следующие обозначения:

$\{R_i\}_{i=1...n}$ - множество рёбер поверхности данной области, n – их количество,

$\{Ra_j\}_{j=1...k}$ - множество угловых рёбер, k – их количество,

$\{C_j\}_{j=1...k}$ - множество концевых точек угловых рёбер с ненулевой высотой,

$S = \bigcup_{i=1}^{n} R_i$ - скелет данной области

Остовом назовём множество точек поверхности растровой области, состоящее из

1. $\mathbf{S} \setminus \bigcup_{j=1}^{k} (Ra_j \setminus C_j)$ - скелета за вычетом угловых рёбер без концевых точек с максимальной высотой.
2. Кратчайших линий на поверхности, соединяющих компоненты связности указанного множества. При этом длина линии понимается как количество точек в ней.

Искомое множество будем обозначать $\widehat{\mathbf{S}}$.

Таким образом, определено связное множество, являющееся компактным однозначным линейным представлением пиксельной области. Далее рассматривается эффективный алгоритм построения этого множества.

3. АЛГОРИТМ ПОИСКА ОСТОВА

С целью экономии памяти компьютера и увеличения скорости выполнения, был разработан индуктивный алгоритм. Свойство индуктивности алгоритма состоит в том, что для анализа входного изображения за один шаг требуется всего одна горизонтальная пиксельная строка. Таким образом, строки обрабатываются последовательно сверху вниз, а соответствующее точное векторное представление (остовы) формируется по ходу движения. Такой подход даёт возможность не хранить всё исходное изображение, а лишь выбирать его построчно. Таким образом, индуктивный алгоритм существенно экономит память компьютера.

3.1 Хорда

Сущность алгоритма состоит в том, что для каждой растровой строки строится некий объект, называемый в данной работе *хордой*. Хорда для данной строки наследуется из предыдущей и лишь итеративно перестраивается с учетом новой информации. Дадим строгое определение введённого этого понятия.

Определение.

Рассмотрим 4-связную растровую область и произвольную горизонтальную строку пикселей этой области. Свяжем с этой строкой систему координат в дискретном пространстве таким образом, что её начало лежит в нижней левой точке данной строки; ось *O'x* сонаправлена с осью *Ox* объемлющего пространства, а *O'y* противонаправлена *Oy* (Рис.5). В выбранной системе координат каждому **x** сопоставим максимальное значение *y*, такое что круг с центром в точке *(x,y)* и радиусом *y* целиком лежит в области. График этой функции назовём *нижней хордой* для данной строки пикселов растровой области.

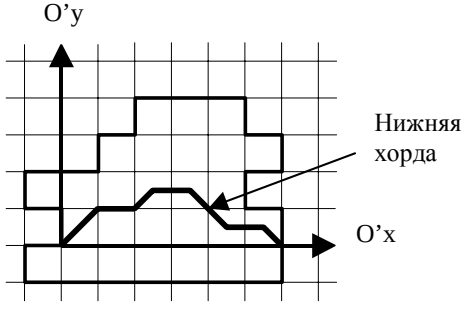

Рис.5. Пример нижней хорды

Понятно, что точки, лежащие на нижней хорде данной пиксельной строки являются центрами максимальных кругов, лежащих в области и касающихся своей нижней гранью нижнего ряда точек, принадлежащих данной строке. Таким образом, можно утверждать, что если бы исходная растровая область была бы ограничена снизу рассматриваемой строкой, то точки несущей плоскости, лежащие под нижней хордой, соответствовали бы одной грани поверхности, построенной над областью. Эта грань была бы порождена точками нижней границы, то есть граничными точками данной строки. Соответственно, точки самой хорды соответствовали бы рёбрам поверхности, так как в этих точках расстояния как минимум до двух частей границы, не лежащих на одной прямой, равны.

По аналогии с понятием нижней хорды данной строки, введём понятие верхней хорды.

Определение.

Для данной строки введём связанную систему координат. Её начало положим в верхней левой точке данной строки; ось *O'x* сонаправлена с осью *Ox* несущей плоскости, а *O'y* противонаправлена *Oy* (Рис.6). В выбранной системе координат каждому *x* сопоставим максимальное значение *y*, такое что круг с центром в точке *(x,y)* и радиусом *y* целиком лежит в области и ему не принадлежат точки лежащие левее или правее соответствующих сторон исходной строки. График этой функции назовём *верхней хордой* для данной строки пикселов растровой области.

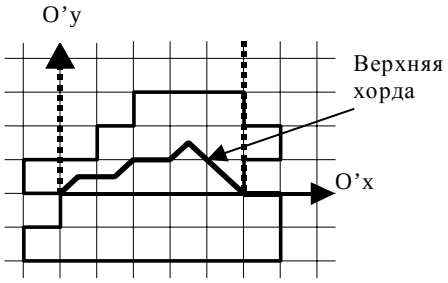

Рис.6. Пример верхней хорды

Нетрудно заметить, что верхняя хорда для данной строки является нижней хордой строки, полученной из предыдущей удалением пикселей лежащих левее и правее соответствующих сторон исходной. На Рис. 6 ограничение верхней строки показано пунктиром.

3.2 Связь хорды и остова

Взаимосвязь построенных хорд с остовом растровой области определяется следующими простыми утверждениями, которые служат основой алгоритма построения остова области на основе анализа хорд последовательных растровых строк.

Утверждение 1. Для любой точки любого ребра поверхности 4-связной растровой области найдётся пиксельная строка такая, что её хорда (нижняя или

верхняя) будет содержать проекцию данной точки на несущую плоскость.

Утверждение 2. Для 4-связной растровой области, если точка *(x,y)* принадлежит хорде (верхней или нижней) пиксельной строки, то точка *(x,y)* с высотой *y* принадлежит поверхности этой области.

Первое определение доказывается перебором возможных случаев, а второе непосредственно следует из определения.

.

Утверждения 1 и 2 по сути определяют связь искомого нами векторного представления (остова) растровой области и понятия хорды для пиксельной строки. Из Утверждения 1 следует, что проекции всех ребер могут быть получены при анализе хорд (верхних и нижних) пиксельных строк, а Утверждение 2 дает возможность вычислять высоты уже полученных точек.

3.3 Основные шаги алгоритма

Из вышесказанного следует, что задачей алгоритма является построение хорд для каждой строки и их анализ. Для текущей строки алгоритм выполняет две операции:

1. Анализ нижней хорды предыдущей строки для вычленения фрагментов искомого множества $\tilde{\mathbf{S}}$ (векторного представления), построение верхней хорды данной строки.
2. Построение нижней хорды также с дополнением где необходимо множества $\tilde{\mathbf{S}}$.

Рассмотрим подробнее каждое действие:

Шаг №1. Построение верхней хорды данной строки. На вход алгоритма подаются координаты концов следующего растрового отрезка, соответствующего области. Задача состоит в построении по нижней хорде предыдущей строки верхней хорды данной. Как замечалось в пункте 3.1, такое преобразование можно осуществить ограничением слева и справа точек текущей хорды (Рис. 7), то есть надо лишь найти точки пересечения текущей хорды с линиями ограничения.

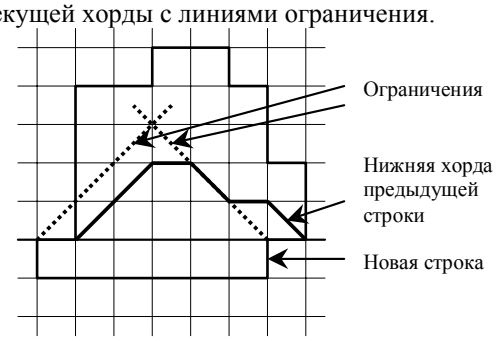

Рис.7. Построение верхней хорды

Анализ принадлежности точек хорды к остову основан на взаимном расположении хорды и новой строки пикселей, а также на рассмотрении точек излома хорды, так как именно они дают нам представление о ребрах – линиях излома поверхности области.

Шаг №2. Построение нижней хорды по верхней. Вторым шагом алгоритма является построение нижней хорды для данной строки руководствуясь информацией о верхней хорде. Так как хорда является полилинией и задаётся множеством точек излома (концов отрезков), то достаточно проследить взаимосвязь этих точек обеих хорд. Точки излома бывают 5-ти типов (рис. 7).

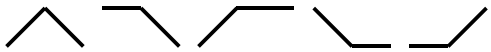

Рис.7. Возможные точки излома.

Излом вида V невозможен, так как круги в точках соседних с точкой излома содержали бы круг с центром в этой точке внутри себя и он бы не был максимальным. Рис. 8 иллюстрирует направления смещения контрольных точек при перестройки хорды. Начальная и конечная точки хорды (y = 0) смещаются соответственно внутрь области с увеличением координаты y. Формируются новые начало и конец с нулевой высотой.

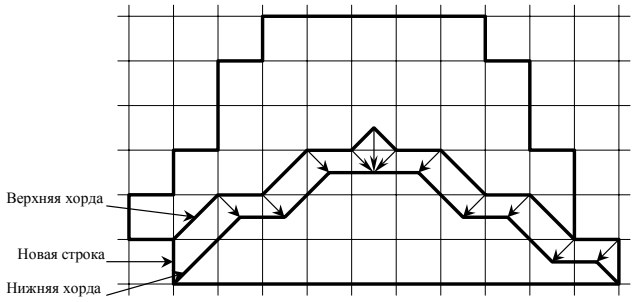

Рис.8. Направления смещений точек излома

Анализ на этом шаге алгоритма состоит в отслеживании слияния нескольких точек излома в одну точку на новой хорде (центр хорды на рис. 23), и дальнейшем исследовании таких слияний.

Таким образом, весь алгоритм поиска векторного представления растровой области состоит из одного цикла, имеющего следующий вид:

Цикл пока не все строки исчерпаны

- Взять следующую строку изображения

- Сформировать верхнюю хорду этой строки, по ходу анализировать взаимное расположение хорды и новой строки, а также точки излома
- Сформировать нижнюю хорду этой строки, анализировать слияния

Конец цикла

Этот алгоритм является индуктивным, то есть анализирует строки растра по одной, пробегая их сверху вниз. Как следствие этого, объем необходимой памяти линейно пропорционален лишь длине растровой строки. К тому же, для одного построения хорды (верхней или нижней) достаточно пробежать все точки излома (концевые точки отрезков, рассматривая хорды как полилинии) слева направо один раз. Таким образом, количество операций, затрачиваемых данным алгоритмом на преобразование изображения, является линейной величиной от количества контрольных точек хорд и линейно пропорционально периметру растровой области, что превосходит по скорости упомянутые во введении существующие алгоритмы.

4. ПОСТОБРАБОТКА ОСТОВА

Для приближения полученного векторного представления (остова) к требованиям тех или иных конечных пользователей, строящееся представление может быть обработано непосредственно в процессе формирования. К постобработке остова можно отнести симплификацию и фильтрацию.

4.1 Симплификация

Симплификацией называется замена линий полученного остова на полилинии (ломаные), состоящие из отрезков большей длины. При этом максимальное отклонение остова от полилинии не должно превышать наперёд заданного числа.

Возможно применения алгоритма симплификации непосредственно при построении остова, и, следовательно, отпадает необходимость в хранении остова в памяти. Результатом преобразования растрового изображения в векторное представление сразу являются полилинии, которые можно использовать в любом системе, работающей с векторной графикой.

4.2 Фильтрация

Как уже отмечалось, основной областью применения алгоритма преобразования растрового изображения в векторный формат являются изображения, полученные с помощью сканера или цифровой камеры. Такие устройства ввода графической информации имеют определённую точность (разрешение). К тому же, исходный материал (например, чертёж) может иметь незначительные искажения или погрешности, которые скажутся при построении векторного представления. Возникает задача об устранении или исправление тех частей полученного остова, которые по некоторому критерию считается ошибочными.

Преобразование полученного векторного представления к виду, более отвечающему пониманию природы исходного изображения (чертеж, текст, и т.п.) называется *фильтрацией*.

Фильтры также могут применяться непосредственно в процессе формирования остова, и, следовательно, конечный пользователь может получить уже исправленное векторное представление.

5. ЗАКЛЮЧЕНИЕ

В статье рассмотрен новый алгоритм получения остова растрового изображения. Предложенный алгоритм является эффективным и оптимальным с точки зрения быстродействия и памяти. Он использует только аддитивную целочисленную арифметику

Быстродействие определяется следующими свойствами:

1. Индуктивность – строки растра выбираются для анализа последовательно по одной сверху вниз
2. Простая целочисленная арифметика
3. Общее количество операций линейно зависит только от периметра области

Так как анализ происходит построчно, то не требуется хранить растровое изображение в памяти компьютера. Память расходуется только на хранение хорд (сравнимо с хранением одной строки растра) и конечного результата – полилиний или цепей, которые по мере построения могут записываться на диск. Таким образом общий объем оперативной памяти может быть ограничен несколькими килобайтами.

Удобство алгоритма для пользователя определяется возможностью обработки остова непосредственно в процессе его формирования, следовательно пользователь может получить уже готовое для использования векторное представление.

Таким образом, представленный алгоритм может быть использован как в качестве независимого дополнение ко многим программным продуктам, так и в качестве встроенного компонента систем работы с векторной графикой, таких как ГИС и системы ОРС.

Простота алгоритма и малый объем необходимой памяти обеспечивает возможность аппаратной реализации и непосредственного использования в средствах ввода изображения, таких как сканер или цифровая камера.

6. ЛИТЕРАТУРА

[1] <Т.Павлидис>. <Алгоритмы машинной графики и обработки изображений, Москва, “Радио и связь”, 1986>.

[2] <Ф.Препарата,М.Шеймос>. <Вычислительная геометрия: Введение, Москва, “Мир”, 1989>.

[3] <Е.В.Шикин,А.В.Боресков>. <Компьютерная графика, Москва, “Диалог-МИФИ”, 1995>.

[4] <Д.Иванов>.<Дипломная работа, Механико-математический факультет МГУ, 1998>

Авторы:

Иванов Денис Владимирович, аспирант механико-математического факультета Московского Государственного Университета

Кузьмин Евгений Павлович, старший научный сотрудник механико-математического факультета Московского Государственного Университета

Адрес: Москва, 119899, Воробьевы горы, МГУ, механико-математический факультет, лаборатория вычислительных методов

Телефон/факс: (095) 319-1378

E-mail: csl@online.ru

D.Ivanov, Ye.Kuzmin

An efficient algorithm for extraction of raster image skeleton.

Raster-to-vector conversion is important problem for wide range of graphic application such as GIS and OCR. This conversion is based on extraction of image skeleton, which preserve all principal information about the image.

Existing methods for skeleton extraction are very memory and time expensive. In this paper a new efficient and optimal algorithm is described. Processing time is linearly depend on area perimeter, and required memory is only few kilobytes. Algorithm uses only simple integer arithmetic and allows hardware implementation.