УДК 515.142.33

БЫСТРЫЙ АЛГОРИТМ ВЕКТОРИЗАЦИИ БИНАРНЫХ ШТРИХОВЫХ ИЗОБРАЖЕНИЙ

М.В. СТЕРЖАНОВ

Белорусский государственный университет информатики и радиоэлектроники
П. Бровки, 6, Минск, 220013, Беларусь

Поступила в редакцию 15 марта 2010

Описывается гибридная методика векторизации штриховых бинарных изображений. Вначале строится гиперграфовое представление каждой связной компоненты изображения. Данная форма представления является компактной и сохраняющей топологию объектов. Граф является нагруженным, его ребра хранят информацию о характеристиках объектов изображения. Предлагается алгоритм построения графовой модели, основанный на кодировании изображения в виде концов серий (RLE). Затем выделяются пути векторизации, применяются быстрые алгоритмы полигональной аппроксимации. Предлагается алгоритм выделения дуг окружностей.

Ключевые слова: бинарный растр, графовая модель, векторизация.

Введение

Под векторизацией понимается процесс перевода растрового изображения (представленного пиксельной матрицей) в векторый формат, состоящий из масштабируемых геометрических примитивов (отрезков, дуг окружностей, кривых Безье). Очевидно, что в данном процессе ошибки неизбежны. В работе [1] авторы отмечают, что эффективность системы распознавания изображений должна измеряться затратами человеческого труда, необходимого на постобработку. Мы придерживаемся этой точки зрения и можем добавить, что эффективность подобных систем должна определяться затраченным операторским временем, которое требуется для получения приемлемого результата. Следовательно, чтобы увеличить производительность подобных систем, необходимо оптимизировать наиболее времязатратные операции.

Большинство известных методик векторизации можно разделить на следующие группы [2], основанные на: преобразовании Хафа, утончении, обходе контуров, графах объектных штрихов, разбиении регулярной сеткой, а также методы разреженного просмотра растра. Основными проблемами при реализации известных алгоритмов векторизации являются ограниченный объем оперативной памяти и требование минимизации времени обработки больших объемов информации. Поэтому обработку зачастую приходится вести фрагментарно, с осуществлением последующей "сшивки" фрагментов.

В данной работе обрабатываются штриховые изображения, объекты которых представлены совокупностью линий, имеющих относительно одинаковую толщину на всем протяжении. Предлагаемый подход является гибридным, т.е. сочетающим в себе различные методики. Вначале каждый объект изображения представляется в виде планарного нагруженного ориентированного псевдографа, в котором все ребра суть прямолинейные отрезки или дуги плоских кривых, а вершины — точки на плоскости, являющиеся концами отрезков или точками сочленения нескольких отрезков. Затем осуществляется нахождение путей векторизации, выполняется векторизация и постобработка. Эффективность подхода достигается за счет быстрого построения сжатой модели растра и быстрого построения промежуточного представления — графовой модели.

Построение графовой модели

Итак, пусть растр представлен в сжатой форме REE (Run Ends Encoding) [3]. Опишем основные шаги алгоритма построения графовой модели. Выделяются связные компоненты(СК) изображения, затем для каждой СК выполняется быстрый алгоритм частичной скелетизации, основной задачей которого является нахождение средних линий объектов прямолинейной формы. Средние линии представляются скелетными кривыми (СКР). Известным недостатком большинства алгоритмов скелетизации является недостаточно точная обработка мест соединений. Совершенно очевидно, что последующая векторизация даст в таком случае ощутимую погрешность. Нами предлагается быстрая обработка "простых" участков и аналитическая — "сложных". Алгоритм состоит из двух проходов. В каждом проходе изображение сканируется вертикальными строками и анализируется связность смежных серий. Последовательность смежных серий, в которой отсутствуют случаи ветвления и слияния серий, будет называться полосой. При работе с каждой скан-строкой изображения добавляются серии к имеющимся полосам. После того, как полосы сформированы, вычисляются их средние линии, которые являются СКР. При сканировании вертикальными строками будут обработаны области, имеющие горизонтальную направленность. Для того, чтобы обработать вертикальные объекты, изображение поворачивается на 90º и выполняется скан-проход алгоритма.

Произведем сравнительный анализ некоторых распространенных алгоритмов утоньшения с предлагаемым алгоритмом частичной скелетизации.

Таблица 1. **Сравнение производительности алгоритмов утоньшения**

Метод	*Min*, мс	*Max*, мс	*Avg*, мс	*Disp*, мс
Хилдича [4]	50	285	146,35	54,32
ZS [5]	571	2662	1342,47	578,17
NWG [6]	59	330	179,32	67,40
NSPTA [7]	213	6277	1067,65	1195,18
MB2 [8]	579	2657	1341,47	576,61
Предлагаемый метод ЧС	8	75	36,10	18,58

Примечания. *Min* — минимальное время утоньшения, *Max* — максимальное время утоньшения, *Avg* — среднее выборочное время утоньшения, *Disp* — квадратный корень среднеквадратичного отклонения времени утоньшения.

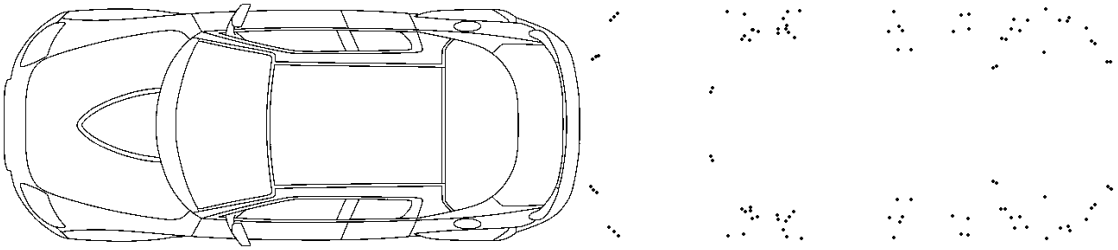
Высокая производительность алгоритма частичной скелетизации обуславливается тем, что он анализирует смежные серии. Алгоритм оставляет необработанными области пересечений и соединений, которые обрабатываются отдельно классическим алгоритмом попиксельного утоньшения. Каждая область соединения (ОС) представлена множеством серий. Из области соединения исходят СКР, аппроксимирующие относительно прямолинейные участки. Для каждой СКР, исходящей из ОС, получим вектор направления, построенный по ее начальным точкам. Найдем точку пересечения векторов направлений ОС и соединим ее отрезками с начальными точками СКР. Пометим точки растра, через которые проходят эти отрезки. Затем применим параллельный алгоритм утоньшения для ОС, который не будет удалять помеченные пиксели. Таким способом обеспечивается корректная обработка соединений. На следующем шаге анализом последовательности серий единичной ширины выделим группы серий, удовлетворяющие условиям СКР. Заменим эти серии с помощью СКР. В данном случае СКР будет являться альтернативным способом представления последовательности пикселей единичной толщины. Перейдем к построению графовой модели. По координатам узловых серий (т.е. серий, которые указывают на СКР) сформируем множество вершин *V*. По точкам СКР построим ребра. Каждая СКР была прикреплена к двум узловым сериям, которые преобразовались в вершины, следовательно, можно найти вершины, из которых исходят ребра. Рассмотрим два смежных ребра. Если они являются коллинеарными (допускается погрешность в 5°), то они могут быть заменены одним ребром.

В результате вышеописанных действий каждая СК изображения представляется нагруженным ориентированным планарным псевдографом, вершинам которого соответствуют концевые и узловые точки отрезков СК, а ребрам — сами отрезки СК, представленные в форме

СКР. Независимые подходы к описанию и построению графовых моделей были предложены в [9–11].

Граф G задается парой $G=(V, E)$, где V — множество вершин. Описание каждой вершины содержит координаты точки, порядок вершины и ее тип (концевая или узловая); E — мультимножество ребер, каждое из которых соединяет две вершины из V, причем изображения ребер из E на плоскости не пересекаются, поэтому (V, E) представляет собой планарный граф. Каждое ребро имеет важные характеристики (например: длина, ширина, элонгация), которые могут быть использованы при последующем создании векторной модели. Т.е. выделенные на этапе векторизации объекты могут быть снабжены соответствующими атрибутами.

Граф может быть преобразован в более компактную форму гиперграфа, гиперребра которого состоят из ребер, соединяющих вершины степени 1 и 2 исходного графа (рис. 1).



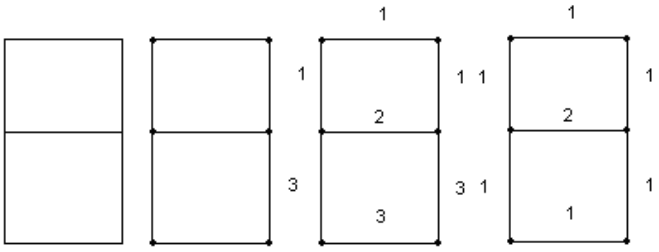
а) *б*)

Рис. 1. Пример графовой модели: *а* — исходное изображение; *б* — узлы гиперграфа

Нахождение путей векторизации

Для последующей постобработки поместим узловые и концевые точки в отдельные массивы [12] (МУТ, МКТ). Следующим шагом является построение векторной модели на основании имеющегося графа. Из имеющегося псевдографа G получим гиперграф GG. На первом шаге процедуры построения выделим гиперребра графа GG, состоящие из ребер графа G, соединяющих вершины графа G степени один и два. Каждое гиперребро имеет две концевые вершины. Из каждой такой вершины гиперребра GE исходит ноль или более одного ребра E, не принадлежащих гиперребру GE. Рассмотрим 2 ребра $E1$ и $E2$, исходящих из вершины степени 3 графа G. Пусть ребра $E1$ и $E2$ принадлежат гиперребрам $GE1$ и $GE2$ соответственно. Если ребра $E1$ и $E2$ являются коллинеарными с некоторой погрешностью (например, 5°), то гиперребра $GE1$ и $GE2$ объединяются.

Введем понятие пути векторизации. Под путем векторизации (ПВ) будем понимать последовательность точек $p_i=(x_i, y_i)$, лежащих на средней линии СК. ПВ=$\{p_0, p_1, \ldots, p_n\}$. Если $p_0=p_n$, то путь является закрытым. Точки ПВ описывают набор отрезков и дуг окружностей. Каждое гиперребро описывает ПВ (рис. 2).



а) *б*) *в*) *г*)

Рис. 2. Пример выделения ПВ: *а* — исходная связная компонента; *б* — полученный граф; *в* — цифрами показаны гиперребра GE на первом шаге построения; *г* — цифрами показаны ПВ

Выделение отрезков прямых

В зарубежной литературе генерализация называется полигонализацией и полигональной аппроксимацией. Полигональная аппроксимация является средством компактного и эффективного представления линий для анализа формы и классификации образов. Она позволяет со-

кратить количество входных данных и сгладить незначительные искажения линий. Оптимальное представление линии должно иметь некоторые из следующих свойств. Во-первых, оно должно хорошо сохранять информацию, т.е. все существенные особенности формы не должны устраняться. Во-вторых, оно не должно использовать значительных объемов памяти. В-третьих, представление должно быть нечувствительно к локальным шумам и преобразованиям, таким как параллельный перенос, поворот, масштабирование и т.д.

Введем понятия сегмента. Под сегментом будем понимать прямолинейный отрезок, полученный путем полигональной аппроксимации точек пути векторизации. Нами реализовано несколько алгоритмов выделения линий. Классическим методом для выделения отрезков является метод генерализации Дугласа–Пекера [13]. Этот метод является итерационной аппроксимацией от концевых точек. Пусть имеется путь векторизации ПВ=$\{p_0, p_1, \ldots, p_n\}$. Рассмотрим процедуру аппроксимации. Построим отрезок p_0p_n. Пусть p_k — самая удаленная от отрезка p_0p_n вершина. Если расстояние от p_k до отрезка p_0p_n меньше заданного порога, то p_0p_n аппроксимирует последовательность. В противном случае разобьем ПВ на две части: $P1=\{p_0, \ldots, p_k\}$ и $P2=\{p_{k+1}, \ldots, p_n\}$. Для каждой части рекурсивно применяется процедура аппроксимации. Возможно использование следующей модификации алгоритма. Пусть имеется путь векторизации ПВ=$\{p_0, p_1, \ldots, p_n\}$. Построим нормированный вектор $\overrightarrow{p_0, p_n}$. Развернем его на 90º против часовой стрелки. Для каждой точки p_i построим вектор $\overrightarrow{p_0, p_i}, i \in 0..n$. Рассчитаем скалярное произведение $(\overrightarrow{p_0, p_i}, \overrightarrow{p_0, p_n})$, которое будет характеризовать отклонение вектора $\overrightarrow{p_0, p_i}$ от $\overrightarrow{p_0, p_n}$. Будем накапливать значения положительного и отрицательного отклонений. После того как отклонения рассчитаны, определяются среднее D_{Aver} и максимальное D_{Max} значения отклонений. Если D_{Aver} превышает пороговую величину среднего отклонения, умноженного на толщину отрезка, то ПВ не может быть представлен одним отрезком. Аналогично, если D_{Max} превышает пороговую величину максимального отклонения, умноженного на толщину отрезка, то ПВ не может быть представлен одним отрезком. В таком случае ПВ разбивается на две равные части, каждая из которых аппроксимируется отдельно.

Штриховые технические изображения содержат большое количество горизонтальных, вертикальных и диагональных линий. Для их выделения на первом, предварительном этапе полигональной аппроксимации, предлагается следующая методика. Вводятся 4 направляющих вектора, описывающих направления линий:

$$\overrightarrow{d_1} = (1,0); \overrightarrow{d_2} = (0,1); \overrightarrow{d_3} = (\frac{\sqrt{2}}{2}, \frac{\sqrt{2}}{2}); \overrightarrow{d_4} = (-\frac{\sqrt{2}}{2}, -\frac{\sqrt{2}}{2}) \quad (1)$$

Последовательно рассматривается каждый направляющий вектор $\overrightarrow{d_i}$. Повернем вектор $\overrightarrow{d_i}$ на 90º против часовой стрелки, обозначим результат через $\overrightarrow{d_i^n}$. В качестве опорной точки b выбирается первая точка ПВ. "Плавающей" точкой f является вторая точка ПВ. По этим двум точкам строится вектор $\overrightarrow{K}$. Вычисляется скалярное произведение векторов $\overrightarrow{K}$ и $\overrightarrow{d_i^n}$. Как известно, скалярное произведение ортогональных векторов равно нулю. Если скалярное произведение равно нулю, в качестве "плавающей" точки выбирается следующая точка ПВ. Если условие не выполняется, процедура повторяется для следующего направляющего вектора.

На следующем этапе полигональной аппроксимации выделяются оставшиеся прямолинейные отрезки. Применяется модификация алгоритма Склански и Гонсалеса последовательного пересечения углов на плоскости. Данная методика является аппроксимацией от одной начальной точки. Алгоритм, предложенный в работе [14], использует следующую процедуру нахождения очередной критической точки. В начале вокруг следующей точки из исходного набора строится окружность радиуса ε и проводятся два луча, выходящие из текущей критической точки, касающиеся данной окружности, образующие угол выбора. Затем, для каждой последующей точки из исходного набора аналогично строится угол, касающийся окружности радиуса ε, проведенной вокруг этой точки, и находится пересечение этого угла с текущим углом выбора. Если данное пересечение пусто, то последняя точка, попавшая в угол выбора, объявляется критической и процедура повторяется для нее. Иначе вычисленное пересечение использу-

ется как угол выбора для следующих точек исходного набора. Нами предлагается следующая эффективная реализация алгоритма. Сначала выбирается опорная точка, т.е. точка, от которой происходит аппроксимация. Опорная точка определяется с помощью анализа массива сегментов, выделенных на первом этапе. Если массив пуст, то в качестве опорной точки *b* полагается первая точка ПВ, в противном случае выбирается точка, не принадлежащая выделенному сегменту. Пусть опорная точка имеет индекс b_i в массиве точек ПВ. "Плавающей" точкой *f* считается следующая точка ПВ, т.е. точка с индексом $f_i=b_i+1$. По этим двум точкам строится текущий вектор направления $\vec{K}$. Повернем вектор $\vec{K}$ на 90º против часовой стрелки, обозначим результат через $\vec{N}$. Построим четыре вектора $\vec{C_i}$, отклоняющихся от $\vec{K}$ на заданную константу(в данной работе 0,5).

$$\vec{C_1} = \vec{K} + (0,5,\ 0); \tag{2}$$

$$\vec{C_2} = \vec{K} + (0,\ 0,5); \tag{3}$$

$$\vec{C_3} = \vec{K} + (-0,5,\ 0); \tag{4}$$

$$\vec{C_4} = \vec{K} + (0,\ -0,5); \tag{5}$$

Рассчитаем скалярное произведение векторов $\vec{C_i}$ и $\vec{N}$, определим минимальное и максимальное значения скалярного произведения. Два вектора $\vec{C_L}$ и $\vec{C_R}$, скалярное произведение с которыми вектора $\vec{N}$ дало экстремальные значения, будут формировать начальный угол выбора. Назовем эти вектора "конусом". На следующей итерации на единицу увеличивается f_i, выбирается новая "плавающая" точка. Строится вектор направления $\vec{K}$, соединяющий опорную и "плавающую" точки. Определяется текущее значение угла выбора. Вычисляется новое значение "конуса" как пересечение векторов текущего "конуса" и "конуса", полученного на предыдущей итерации. Т.е. на каждой итерации "конус" будет сужаться. Алгоритм продолжает свою работу до тех пор, пока "конус" не вырождается в прямую. Следует отметить, что данный алгоритм является эффективным, так как его трудоемкость линейна относительно количества исходных точек.

Выделение дуг окружностей

Согласно имеющимся исследованиям [15], существуют два основных семейства алгоритмов выделения дуг окружностей. Первое семейство основано на модификациях преобразования Хафа (ПХ). Данные алгоритмы работают с пикселами изображения напрямую. Они достаточно хорошо изучены и обладают высоким качеством распознавания при работе с зашумленными изображениями. Однако использование ПХ требует большого количества вычислений, имеется квадратичная зависимость трудоемкости от точности распознавания, серьезным недостатком является недостаточно точное нахождение центра и начальных точек дуги окружности. Второе семейство алгоритмов работает с последовательностью сегментов, являющихся результатом процедуры полигональной аппроксимации. С помощью анализа смежных сегментов определяются параметры дуги окружности.

Итак, пусть результатом имеется результат генерализации, представленный сегментами *S1, S2, …, Sn.* Для правильного выделения дуги окружности требуется, чтобы сегменты полилинии достаточно точно аппроксимировали дугу. Во-первых, полилиния должна иметь одинаковую кривизну, т.е. каждый последующий сегмент должен поворачивать по отношению к предыдущему либо по часовой, либо против часовой стрелки. Во-вторых, все сегменты полилинии должны иметь примерно одинаковую толщину. В-третьих, точки средней линии найденных сегментов должны принадлежать дуге окружности, т.е. должны выполняться условия:

$$\left|r - dp_i\right| \le \frac{w}{2};\ i = 0,1,...,N, \tag{6}$$

$$\left|r - de_i\right| \le \frac{w}{2}; i = 0,1,...,N; \quad (7)$$

где r — радиус дуги; w — толщина полилинии; dp_i — расстояние от центра дуги до i-й узловой точки полилинии; de_i — расстояние от центра дуги до i-го сегмента полилинии; N — число сегментов полилинии.

Из Евклидовой геометрии известно, что три неколлинеарные точки X, Y, Z на плоскости однозначно определяют окружность. Центр окружности U будет находиться на пересечении перпендикуляров, опущенных на середины отрезков XY и YZ. Рассмотрим пару смежных сегментов S_iS_{i+1}. В качестве точки X возьмем первую точку сегмента $S_{i,}$, в качестве точки Z возьмем последнюю точку сегмента S_{i+1}. Таким образом сформирован набор точек ПВ, который может быть аппроксимирован дугой окружности. В качестве точки Y выбирается точка ПВ, находящаяся в массиве ПВ на равном расстоянии от X и Z. Вычисляется центр дуги и радиус. Рассчитывается отклонение точек сегментов S_iS_{i+1} от уравнения дуги окружности. Если все условия выполнены, то дуга найдена. Затем итеративно осуществляются попытки продолжения дуги. Так как дуга имеет две концевые точки, существуют два направления расширения дуги. В каждой итерации к сегментам, образовавшим дугу, добавляется новый сегмент. Определяется новый центр и радиус, осуществляется проверка корректности.

Полученные из сегментов отрезки и дуги окружностей получают характеристику ширины ребра графа G, из анализа которого они были получены.

В качестве постобработки предлагается стыковка отрезков в местах соединений. Это осуществляется анализом МУТ и МКТ.

На рис. 3 приведен пример работы алгоритма.

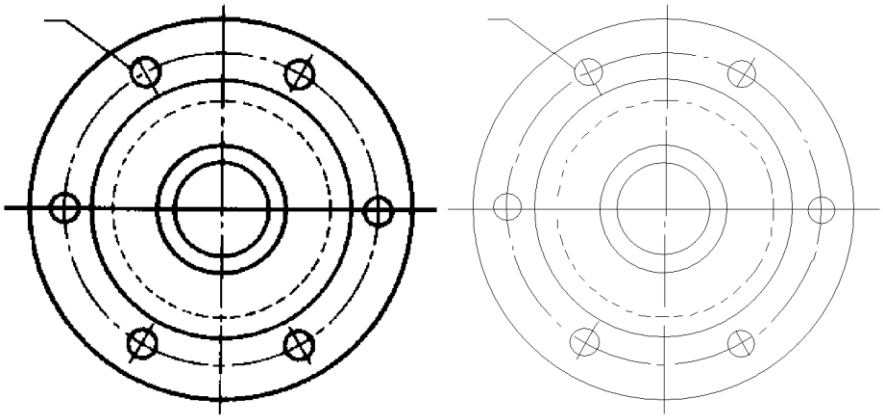


Рис. 3. Пример векторизации: исходное изображение и результат векторизации

Заключение

Предложен гибридный метод векторизации бинарного растра, выделяющий отрезки прямых и дуги окружностей. Мы предлагаем использование графовой модели в роли промежуточного представления растра. Данный вид представления является компактным средством хранения и описания структуры изображения. Применение сжатой модели растра позволяет производить обработку чертежей большого размера. Благодаря использованию алгоритма частичной скелетизации построение скелета осуществляется быстрее, чем при использовании классических методов попиксельного анализа. Также решается проблема обработки соединений. Благодаря аналитической обработке мест соединений и отсутствию разрывов в полученном графе последующее выделение дуг окружностей является менее трудоемким и более точным по сравнению с методами разреженного просмотра, допускающими разрывы в генерируемых сегментах. Кроме того, в процессе построения графовой модели вычисляются важные геометрические, топологические и текстурные характеристики объектов. Объекты полученной векторной модели получают рассчитанные характеристики графовой модели. Достоинствами предлагаемого подхода являются простота реализации и достаточно высокое быстродействие.

Недостатком подхода является неполное описание площадных объектов. Эффективность предлагаемого подхода подтверждена экспериментально.

FAST VECTORIZATION ALGORITHM OF LINE IMAGE DRAWINGS

M.V. STERJANOV

Abstract

Hybrid method for raster line image vectorization is presented. First, a hypergraph representation for every connected component is obtained. This form of representation is compact and topology preserving. This form of compression preserves the topology. Fast algorithm for graph model construction is suggested. The algorithm is based on run length encoding of the image. Then vectorization paths are created. Segments are extracted using fast polygonal approximation algorithm. Arc segmentation algorithm is also presented.

Литература

1. *Chhabra A., Phillips I.* // Graphics recognition–algorithms and systems. Proc. of the 2nd Workshop, Lecture Notes in Computer Science. 1998. P. 390–410.
2. *Liu W., Dori D.* // Pattern Analysis and Applications. 1999. No. 1. P. 10–21.
3. *Burge M., Kropatsch W.G.* // Computing. Vol. 62, Issue 4. 1999. P. 355–368.
4. *Hilditch C.J.* // Machine Intelligence IV, Meltzer, B., Michie, D. (Eds.), University Press, Edinburgh. 1969. P. 403–420.
5. *Zhang T.Y., Suen C.Y.* // Comm. ACM. 1984. Vol. 27. P. 236–239.
6. *Nagendraprasad M.V., Wang P.S.P., Gupta A.* // Digital Signal Processing. 1993. Vol. 3. P. 97–102.
7. *Ng G.S., Zhou R.W., Quek C.* // IEEE Trans. on System Man and Cybernetics. 1994. Vol. 3. P. 80–101.
8. *Bernard T.M., Manzanera A.* // Proc. 10th Int. Conference on Image Analysis and Processing (ICIAP'99). 1999. P. 215–225.
9. *Костюк Ю.Л., Новиков Ю.Л.* // Вестник ТГУ. 2002. № 275. С. 153–160.
10. *Kropatch W.G. et al.* Graphs in image analysis // Digital Image analysis. Selected techniques and applications. New York, 2001. P. 6, P. 179–198.
11. *Di Zenzo S., Cinque L., Levialdi S.* // Pattern Analysis and Machine Intelligence, IEEE Transactions on PAMI. 1996. Vol. 18, № 1. P. 83–89.
12. *Семенков О.И., Абламейко С.В., Берейшик В.И., Старовойтов В.В.* Обработка и отображение информации в растровых графических системах. Минск, 1989.
13. *Douglas D.H., Peucker T.K.* // The Canadian Cartographer. 1973. Vol. 10, № 2. P. 112–122.
14. *Sklansky J., Gonzalez V.* // Pattern Recognition. 1980. Vol.12. P. 327–331.
15. *Wenyin L., Dori D.* // IEEE Trans. on Pattern Analysis and Machine Intelligence. 1998. Vol. 20, № 4. P. 424–431.