*УДК 621.391*

# ПОИСК ЛОКАЛЬНЫХ ЭКСТРЕМУМОВ ПОЛУТОНОВЫХ ИЗОБРАЖЕНИЙ НА ОСНОВЕ ЦЕНТРАЛЬНО-СИММЕТРИЧНОГО СКАНИРОВАНИЯ

А.Т. НГУЕН, В.Ю. ЦВЕТКОВ

*Белорусский государственный университет информатики и радиоэлектроники, Республика Беларусь*

*Поступила в редакцию 18 ноября 2018*

**Аннотация.** Предложен алгоритм поиска однопиксельных экстремумов полутоновых изображений на основе центрально-симметричного сканирования. Показано, что алгоритм работает значительно быстрее, чем лучшие известные алгоритмы обнаружения ключевых точек изображений.

*Ключевые слова:* поиск локальных экстремумов, центрально-симметричное сканирование.

## Введение

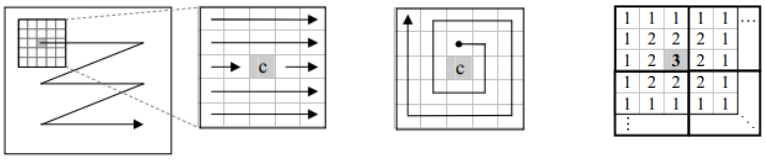
Поиск локальных экстремумов является базовой операцией для множества задач обработки изображений. Известен алгоритм NMS (Non-maximum Suppression – подавление немаксимальных значений), который первоначально использовался для уменьшения длительности откликов при детектировании тонких линий [1]. Алгоритм NMS является одномерным (1-D) и работает перпендикулярно к краям. В работе [2] предложен способ модификации алгоритма NMS для определения ключевых точек (реперов) изображения в двухмерном пространстве пикселей изображения (2-D). Ключевые точки выбираются как локальные максимумы изображения над некоторой окрестностью. Этот подход к обнаружению углов был принят многими детекторами ключевых точек [3–5]. При исследовании эффективности поиска локальных экстремумов берется ориентир на алгоритмы, требующие минимального использования памяти.

Известные алгоритмы NMS требуют фиксированного количества сравнений на пиксель независимо от размера окрестности исключения. Одномерный максимальный фильтр, например, требует трех сравнений на пиксель [6–8]. При последовательной реализации двухмерный максимальный фильтр использует шесть сравнений на пиксель. В работе [9], опубликованной в 2006 году, предложен алгоритм разбиения блоков, который уменьшает количество сравнений до 2,39 на пиксель. Однако, 20-ю годами ранее, в работе [10] предложен алгоритм, имеющий аналогичную вычислительную сложность. В любом случае для большинства известных алгоритмов поиска локальных экстремумов необходимо выполнять более двух сравнений на пиксель. В работе [1] модифицированы алгоритмы, предложенные в [9, 10] с целью уменьшения количества сравнений на пиксель до значения менее двух.

## Алгоритмы поиска локальных экстремумов на изображениях

Прямой подход к поиску экстремумов в двухмерном массиве представлен на рис. 1, *а*. Пиксели исходного изображения анализируются в порядке растрового сканирования (слева направо, затем сверху вниз). Каждый анализируемый пиксель сравнивается с другими пикселями в своей окрестности размером $5\times5$ пикселей также в порядке растрового сканирования [9]. Центральный пиксель C не является максимальным, если найден любой более значимый или равный соседний пиксель. Затем алгоритм переходит к следующему пикселю в строке сканирования. Прямой подход требует $(2n+1)\times(2n+1)/2$ сравнений на пиксель для $(2n+1)\times(2n+1)$-окрестности. Наилучшему варианту сооветствует ситуация, когда вектор

интенсивности меняется на противоположный. При этом прямой подход требует только одного сравнения на пиксель. В среднем, однако, данный подход требует $O(n)$ сравнений на пиксель.




*а* *б* *в*

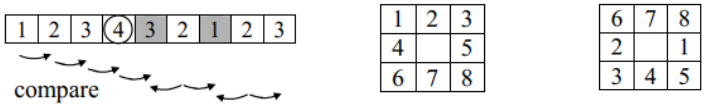
Рис. 1. Представление способов сканирования изображений: *а* – растровое сканирование; *б* – спиральное сканирование; *в* – Block partitioning

Сложность алгоритма растрового сканирования может быть значительно уменьшена путем анализа соседних пикселей в другом порядке. В работе [10] представлен такой алгоритм с локальным спиральным порядком (рис. 1, *б*). Сначала, в результате сравнения, центральный пиксель, возможно, будет локальным максимумом в $3\times3$-окрестности. Затем он проверяется в большей окрестности. Так как количество локальных максимумов в $3\times3$-окрестности в изображении обычно невелико ($\le 25$ % от общего количества пикселей), алгоритм спирального порядка быстро находит любые немаксимальные значения, пропускает их и переходит на следующий пиксель. Число локальных максимумов в окрестности с размером $(2n+1)\times(2n+1)$ пикселей также быстро уменьшается, поскольку размер окрестности увеличивается. В результате вычислительная сложность этого алгоритма примерно постоянна (не более 5 сравнений на пиксель для обнаружения в $3\times3$-окрестности немаксимальных пикселов) независимо от размера окрестности.

В работе [9] представлен эффективный алгоритм NMS, который требует 2,39 сравнений на пиксель в среднем и 4 сравнения на пиксель в худшем случае. Они отметили, что максимальный пиксель в окрестности размером $(2n+1)\times(2n+1)$ пикселя также является максимальным для любого окна размером $(n+1)\times(n+1)$ пикселей. Входное изображение разбивается на неперекрывающиеся блоки размера $(n+1)\times(n+1)$ пикселей и локальный максимум каждого блока детектируется (рис. 1, *в* иллюстрирует это для $n=2$). Затем для максимального размера блока $(2n+1)\times(2n+1)$ пикселей за исключением охватывающего блока $(n+1)\times(n+1)$ пикселей. Используя только одно сравнение на пиксель, шаг разбиения блока уменьшает количество локальных максимумов с фактором $(n+1)\times(n+1)$. В результате метод достаточно эффективен для больших размеров окрестностей. Решение уменьшить количество дополнительных сравнений на одного кандидата до $2+O(1/n)$, значительно увеличивает сложность алгоритма и использование памяти.

Метод NMS для окрестности $3\times3$ пикселей часто решается с помощью математической морфологии [11, 12], в результате чего входное изображение сравнивается с его дилатацией серого цвета. Пиксели, где два изображения равны, соответствуют локальным максимумам. Однако математическая морфология не возвращает строгие локальные максимумы, где центральный пиксель строго больше, чем все соседние пиксели. С точки зрения вычислительной сложности морфология также неэффективна – реализация дилатации для $3\times3$-окрестности полутонового изображения требует шести сравнений на пиксель [6, 7].

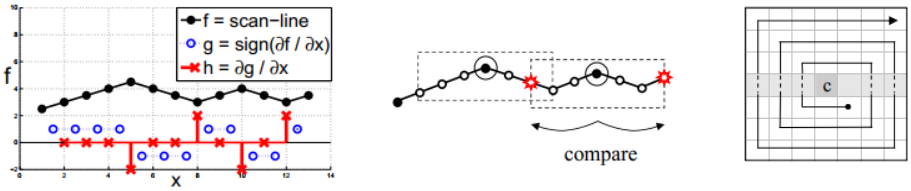
В [13] предложен алгоритм $3\times3$ сканирующей линии для NMS, который требует не более 2 сравнений на пиксель. Алгоритм сначала ищет одномерные локальные максимумы вдоль линии сканирования. Затем каждый максимальный уровень сканирования сравнивается с соседними пикселями в соседних строках. Две двоичные маски сохраняются для текущей и следующей строк сканирования в буфере. По мере обработки нового центрального пикселя соседние ему пиксели маскируются, если они меньше центрального пикселя. Маскированные пиксели будут пропущены, когда наступит их очередь обработки (рис. 2). В результате этот алгоритм NMS для окрестности $3\times3$ пикселей требует не более двух сравнений на пиксель.




*а* *б* *в*

Рис. 2. Маски сканирующей линии $3\times3$ – окрестности: *а* – 1-D Non-maximum Suppression [13]; *б* – растровое сканирование; *в* – Scan-line [13]

Алгоритм сканирующей линии для $3\times3$ -окрестности может быть расширен до блоков $(2n+1)\times(2n+1)$ пикселей при $n\geq1$ [13]. Предположим, что максимумы $(2n+1)$-окрестности на текущей линии сканирования расположены так, как показано на рис. 3, *б* (пиксели в окружностях). Эти 1-D максимумы служат кандидатами на двумерные максимумы. Каждый кандидат проверяется на экстремум в $(2n+1)\times(2n+1)$-окрестности в спиральном порядке, аналогичном методу Forstner [10]. При этом соседние пиксели, расположенные на одной линии сканирования, уже были сопоставлены и потому могут быть пропущены (серые пиксели на рис. 3, *в*). Это приводит к тому, что максимум для $2n\times(2n+1)$-соседей сравнивается с каждым кандидатом. В результате среднее количество сравнений на один кандидат намного меньше благодаря порядку перемещения по спирали.




*а* *б* *в*

Рис. 3. Сканирующий алгоритм NMS для $(2n+1)\times(2n+1)$-окрестности ( $n=3$ ): *а* – 1-D обнаружение пиков; *б* – 1-D исключение не-максимумов; *в* – спиральное сканирование

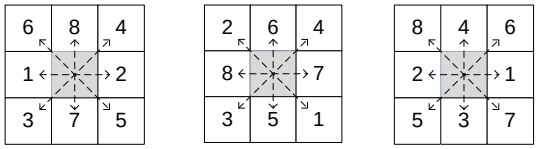
Обнаружение максимумов в $(2n+1)$ – окрестности на одномерной сканирующей линии функции $f$ подробно показано на рис.3 а. Если $g$ – знак конечной разности функции $f$, значение $g$ равно либо –1, 0 или 1 в зависимости от локального наклона $f$. Следовательно, гонечная разность $h$, равна –2 на локальных пиках, +2 в локальных желобах и 0 в другом месте. Таким образом, для обнаружения 1-D пика и минимума требуется только одно сравнение на пиксель. Затем каждый 1-D экстремум сравнивается с его участком в $(2n+1)$ – окрестности со знанием экстремума детектора $h$. Соседние пиксели, которые находятся на последовательном склоне вниз от локального пика, т. е. $\{x\,|\,h(x)=0\}$ , по определению меньше, чем текущий пик, поэтому их не нужно повторно сравнивать. Пиксели, расположенные вне закрывающих впадин текущего пика, требуются дополнительного сравнения. Число дополнительных сравнений для получения максимумов $(2n+1)$-окрестности из исходного списка максимумов $3\times3$-окрестности очень мало для гладкой функции $f$.

Недостатками рассмотренных выше алгоритмов являются низкая скорость поиска, наличие ошибок обнаружения экстремумов на границах блоков изображения, пропуск локальных минимумов. В этой связи актуальной является задача поиска всех локальные однопиксельных экстремумов (как максимумов, так и минимумов) на изображении с низкой вычислительной сложностью без использования дополнительной памяти. Предлагаемый алгоритм позволяет быстро найти все локальные однопиксельные экстремумы.

## Алгоритм поиска локальных экстремумов полутоновых изображений

Для поиска на изображении однопиксельных экстремумов предлагается алгоритм на основе центрально-симметричного сканирования (SSEF – SymmetricScan-based Extreme Finding). Сущность алгоритма состоит в выборе пикселей изображения в порядке строчной развертки, оценке распределения яркости в окрестностях размером $3\times3$ пикселей, фиксировании центрального пикселя в качестве экстремума, если все пиксели в его окрестности

имеют меньшие или большие значения. Предлагаемый алгоритм отличается от известных алгоритмов поиска однопиксельных экстремумов порядком выборки пикселей в окрестности анализируемого на экстремум пикселя, который осуществляется центрально-симметрично – поочередно с различных направлений относительно центра области анализа (рис. 4). Центральный пиксель *C* не является максимальным, если найден любой более значимый или равный ему пиксель. Немаксимальный пиксел затем передается на этап поиска локальных минимумов. Если найден любой менее значимый или равный пиксель, то центральный пиксель *C* также не является минимальным и алгоритм переходит на следующий пиксель по порядку сканирования изображения. Процедура поиска продолжается до тех пор, пока все пиксели не будут обработаны.




*а* *б* *в*

Рис. 4. Виды центрально-симметричной окрестности размером $3\times3$ пикселя

Алгоритм поиска состоит из следующих шагов

1. Инициализация. На данном шаге осуществляется буферизация исходного изображения $I=\|i(y,x)\|_{(y=\overline{0,Y-1},x=\overline{0,X-1})}$ размером *YX* пикселей. Формируется матрица разметки экстремумов $E=\|e(y,x)\|_{(y=\overline{0,Y-1},x=\overline{0,X-1})}$, элементы которой определяются с помощью выражений $e(y,x)\leftarrow1$ при $y=\overline{0,Y-1},\ x=\overline{0,X-1}$.

Формирование вектора индексов центрально-симметричного сканирования с помощью выражения $RC=\|rc(k)\|_{(k=\overline{0,7})}$, элементы которого определяются следующим образом: $rc(k)\leftarrow[0\ -1;\ 0\ \ 1;\ 1\ -1;-1\ \ 1;\ 1\ \ 1;-1\ -1;-1\ \ 0]$ при $k=\overline{0,7}$ (рис.4 а).

2. Начало цикла поиска локальных однопиксельных экстремумов

2.1. Поиск локальных однопиксельных максимумов. Осуществляется формирование матрицы разметки значений максимумов $E=\|e(y,x)\|_{(y=\overline{0,Y-1},x=\overline{0,X-1})}$ с помощью выражения

$$\forall(e(y,x)=1)\Rightarrow$$
$$\Rightarrow e(y,x)=\begin{cases}0\ \text{ при }\exists k(k\in[0,7])(i(y+rc(k,1),x+rc(k,2))\le i(y,x))\\1\ \text{иначе (есть местоположение максимумов)}\end{cases}\qquad(1)$$

при $y=\overline{0,Y-1},\ x=\overline{0,X-1}$.

2.2. Поиск локальных однопиксельных минимумов. Осуществляется формирование матрицы разметки значений минимумов $E=\|e(y,x)\|_{(y=\overline{0,Y-1},x=\overline{0,X-1})}$ с помощью выражения

$$\forall(e(y,x)=0)\Rightarrow$$
$$\Rightarrow e(y,x)=\begin{cases}2\text{ при }\exists k(k\in[0,7])(i(y+rc(k,1),x+rc(k,2))\ge i(y,x))\\0\ \text{иначе (есть местоположение минимумов)}\end{cases}\qquad(2)$$

при $y=\overline{0,Y-1},\ x=\overline{0,X-1}$

2.3. Проверка условия окончания цикла: $y\le Y$ и $x\le X$. Если выполняется условие, то осуществляется переход к шагу 2.1. Если это условие не выполняется, то осуществляется выход из алгоритма.

Вычислительная сложность алгоритма определяется числом вещественных операций сложения (алгоритм имеет нулевую вычислительную сложность умножения) с помощью выражения

$$A_{\text{слж}}=2\times CPP\times Y\times X.\qquad(3)$$

где $CPP$ – число сравнений на пиксель (Comparisons Per Pixel), $Y, X$ – размеры исходного изображения.

В результате выполнения данного алгоритма формируется матрица разметки локальных экстремумов, значение каждого элемента которой указывает на экстремум изображения (значение 1 – максимум, значение 0 – минимум), которому принадлежит пиксель разметки изображения с соответствующими координатами. Эти данные используются для последующей обработки изображений.

## Оценка эффективности алгоритма поиска локальных экстремумов на основе центрально-симметричного сканирования

Выполнено сравнение предложенного алгоритма с некоторыми известными алгоритмами, описанными в среде Matlab и Matlab ®Image Processing Toolbox (R2015a): imdilate и imerode, которые использовались для реализации морфологии [11, 12]. Для этого эксперимента были выбраны три изображения размером $512\times512$ пикселей: Lena, Barbara и Airfield (рис. 5).

*а* *б* *в*

Рис.5. Тестовые полутоновые изображения: *а* – «Lena»; *б* – «Barbara»; *в* – «Airfield»

В таблице приведены значения экстремумов, экспериментально установленные для трех полутоновых изображений различных типов (Lena – студийное низкочастоное с размером $512\times512$ пикселей с преобладанием участков с плавным изменением яркости; Barbara – студийное высокочастотное с размером $512\times512$ пикселей с преобладанием участков с резким измениением яркости; Airfield – аэроизображение с размером $512\times512$ пикселей), значения времени поиска и числа сравнений на пиксель в среде Matlab 2015a, экспериментально полученные для алгоритмов Straightforward, Forstner [10], Neubeck [9], Scanline3x3 [13], Scanline-spiral [13] и предложенного SSEF. Количество сравнений на пиксель и средняя продолжительность выполнения были установлены для каждого тестового изображения в системе Intel Core i5 2.3 ГГц с 4 ГБ ОЗУ.

**Результат поиска экстремумов изображений с размером $512\times512$**

| Методы | Изображение | Число экстремумов | | | Время, с | Число сравнений | Память |
|---|---|---|---|---|---|---|---|
| | | Сумма | Поиск | Ошибка | | | |
| Straightforward | Lena | 33078 | 23404 | 0 | 0,241 | 9,66 | $O(1)$ |
| | Barbara | 32322 | 26233 | 0 | 0,240 | 9,81 | |
| | Airfield | 47414 | 33876 | 0 | 0,235 | 9,86 | |
| Forstner [10] | Lena | 33078 | 23404 | 0 | 0,107 | 4,47 | $O(1)$ |
| | Barbara | 32322 | 26233 | 0 | 0,117 | 4,63 | |
| | Airfield | 47414 | 33876 | 0 | 0,116 | 4,85 | |
| Neubeck [9] | Lena | 33078 | 26395 | 1799 | 1,420 | 3,22 | $O(1)$ |
| | Barbara | 32322 | 27709 | 1141 | 1,441 | 3,26 | |
| | Airfield | 47414 | 35097 | 977 | 1,415 | 3,45 | |
| Scanline3x3 [13] | Lena | 33078 | 25401 | 602 | 1,043 | < 4 | $2\times O(1+2/X)$ |
| | Barbara | 32322 | 27626 | 548 | 0,993 | < 4 | |
| | Airfield | 47414 | 34573 | 318 | 1,113 | < 4 | |
| Scanline - spiral order [13] | Lena | 33078 | 23304 | 0 | 0,625 | < 4 | $2\times O(1+2/X)$ |
| | Barbara | 32322 | 25894 | 0 | 0,577 | < 4 | |
| | Airfield | 47414 | 33360 | 0 | 0,762 | < 4 | |
| Предложенный (SSEF) | Lena | 33078 | 23404 | 0 | 0,067 | 4,21 | $O(1)$ |
| | Barbara | 32322 | 26233 | 0 | 0,072 | 4,48 | |
| | Airfield | 47414 | 33876 | 0 | 0,075 | 4,52 | |

## Заключение

Предложен быстрый алгоритм поиска экстремумов изображения на основе центрально-симметричного сканирования. Установлено, что алгоритм SSEF по сравнению с Forstner [10] обеспечивает уменьшение в 1,05 раза числа сравнений на пиксель и повышение в 1,5 раза скорости поиска, по сравнению с алгоритмом Straightforward – уменьшение в 2,2 раза числа сравнений на пиксель и повышение в 3,5 раза скорости поиска, по сравнению с алгоритмом Scanline3x3 [13] – увеличение в 1,05 раза числа сравнений на пиксель, но повышение в 15 раз скорости поиска, по сравнению с алгоритмом Scanline-spiral [13] – увеличение в 1,05 раза числа сравнений на пиксель, но повышение в 10 раз скорости поиска. Алгоритм Neubeck [9] обеспечивает меньшее число сравнений на пиксель, однако имеет ошибку поиска на границах блоков разбиения. Алгоритм SSEF не использует дополнительную память, что важно для многих задач компьютерного зрения, таких как обнаружение объектов и углов. Предложенный алгоритм также может быть модифицирован для обработки данных любого измерения в режиме реального времени.

## SEARCH OF LOCAL EXTREMUMS OF HALF-TONE IMAGES BASED ON CENTRAL SYMMETRIC SCANNING

NGUYEN ANH TUAN, V.Yu. TSVIATKOU

**Abstract.** An algorithm for searching for single-pixel extremes of halftone images based on centrally symmetric scanning is proposed. It is shown that the algorithm works much faster than the best known algorithms for detecting key points of images.

*Keywords:* local extremum search, centrally symmetric scanning.

## Список литературы

1. Rosenfeld A., Kak A. Digital Picture Processing. Academic Press, 1976.
2. Kitchen L., Rosenfeld A. // Pattern Recognition Letters. 1982. Vol. 1. P. 92–102.
3. Harris C., Stephens M. // Proc. of the Fourth Alvey Vision Conference. 1988. P. 147–151.
4. Lowe D. // IJCV. 2004. Vol. 60. P. 91–110.
5. Mikolajczyk K., Schmid C. // IJCV. 2004. Vol. 60. P. 63–86.
6. Van Herk M. // Pattern Recognition Letters. 1992. Vol. 13. P. 517–521.
7. Gil J., Werman M. // IEEE Trans. on PAMI. 1993. Vol. 15. P. 504–507.
8. Coltuc D., Bolon P. // Proc. Of EUSIPCO. 2000. P. 2425–2428.
9. Neubeck A., Van Gool L. // Proc. of ICPR. 2006. Vol. 3. P. 850–855.
10. Forstner W., Gulch E. // Proc. of Intercommission Conf. on Fast Processing of Photogrammetric Data. 1987. P. 281–305.
11. Soille P. / Morphological Image Analysis: Principles and Applications. Springer. 2006.
12. Р. Гонсалес, Вудс Р. Цифровая обработка изображений. М., 2005.
13. Tuan Q. Pham // Advanced Concepts for Intelligent Vision Systems (ACIVS). 2010. Vol. 12. P. 438–451.