УДК 004.932+004.258
https://doi.org/10.38086/2522-9613-2020-2-43-50

Использование стеков для сегментации изображений на основе выращивания областей

В. Ю. Цветков, д. т. н., доцент, заведующий кафедрой инфокоммуникационных технологий

E-mail: vtsvet@bsuir.by

УО «Белорусский государственный университет информатики и радиоэлектроники», ул. П. Бровки, д. 6, 220013, г. Минск, Республика Беларусь

Аннотация. Цель работы заключается в сравнительной оценке размеров FIFO- и LIFO-стеков, требуемых для сегментации изображений на основе выращивания областей. В процессе сегментации в стеки помещаются координаты (y, x) пикселей, которые необходимо присоединить к выращиваемой области. Размер стека, необходимый для хранения координат, зависит от структуры изображения и не может быть определен до сегментации. Для исключения переполнения стека его размер определяется для условий максимальной загрузки, когда изображение содержит единственную максимальную область. В этом случае размер стека равен размеру изображения. Такой подход не учитывает процесс его выгрузки и ведет к перерасходу памяти. В работе получены выражения, позволяющие повысить точность определения размеров FIFO- и LIFO-стеков, необходимых для хранения координат смежных пикселей в алгоритме сегментации на основе выращивания областей в условиях максимальной загрузки. При этом учтены начальное положение точки роста области и направление выборки смежных пикселей в окне сканирования. Сравнительная оценка размеров стеков, необходимых для сегментации изображений, показала, что использование FIFO-стека предпочтительнее, чем LIFO-стека, и ведет к существенной экономии памяти.
Ключевые слова: FIFO-стек, LIFO-стек, сегментация изображений, выращивание областей, размер стека
Для цитирования: Цветков, В. Ю. Использование стеков для сегментации изображений на основе выращивания областей / В. Ю. Цветков // Цифровая трансформация. – 2020. – № 2 (11). – С. 43–50. https://doi.org/10.38086/2522-9613-2020-2-43-50

Using Stacks for Image Segmentation Based on Region Growing

V. Yu. Tsviatkou, Dr. Sc. (Technology), Associate Professor, Head of Department of Infocommunications

E-mail: vtsvet@bsuir.by

Belarusian State University of Informatics and Radioelectronics, 6 P. Brovka Str., 220013 Minsk, Republic of Belarus

Abstract. The aim of the work is to comparatively evaluate the sizes of FIFO and LIFO stacks required for image segmentation based on growing regions. The coordinates (y, x) of the pixels that need to be attached to the cultivated area are placed in stacks during the segmentation process. The size of the stack needed to store the coordinates depends on the structure of the image and cannot be determined before segmentation. To avoid stack overflow, its size is determined for maximum load conditions when the image contains a single maximum area. In this case, the stack size is equal to the image size. This approach does not take into account the process of stack unloading and leads to memory overrun. Expressions are obtained in the paper that allow one to increase the accuracy of determining the sizes of FIFO and LIFO stacks necessary for storing the coordinates of adjacent pixels in a segmentation algorithm based on growing regions under maximum load conditions. In this case, the initial position of the region growth point and the direction of the selection of adjacent pixels in the scanning window are taken into account. A comparative assessment of the stack sizes required for image segmentation showed that using the FIFO stack is preferable to the LIFO stack and leads to significant memory savings.
Key words: FIFO stack, LIFO stack, image segmentation, region growing, stack size
For citation: Tsviatkou V. Yu. Using Stacks for Image Segmentation Based on Region Growing. *Cifrovaja transformacija* [Digital transformation], 2020, 2 (11), pp. 43–50 (in Russian). https://doi.org/10.38086/2522-9613-2020-2-43-50

Введение. Одна из основных проблем организации обработки данных в вычислительной системе – выделение памяти. Её недостаточное количество приводит к потерям данных и сбоям. Избыточное же выделение памяти ведет к росту стоимости обработки данных. В обработке изображений данная проблема обостряется в связи с тем, что промежуточные результаты многих алгоритмов занимают существенно больше памяти в сравнении с исходными данными. Примером могут служить алгоритмы сегментации изображений [1–4]. Сегментация приводит к разделению изображения на области с одинаковыми или схожими свойствами. Простейшие алгоритмы сегментации основаны на выращивании областей в окрестности предварительно выделенных начальных пикселей роста и используют стеки для хранения координат смежных пикселей, присоединяемых к выращиваемой области [2, 5–7]. Стеки могут быть организованы по принципу FIFO (первым зашел – первым вышел) или LIFO (последним зашел – первым вышел) [8–10]. Они отличаются порядком выборки данных и, соответственно, размером, необходимым для хранения координат пикселей при максимальной загрузке. Целью работы является сравнительная оценка размеров FIFO- и LIFO-стеков, требуемых для алгоритма сегментации изображений на основе выращивания областей.

Постановка задачи. В результате сегментации изображения $I = \|i(y,x)\|_{(y=\overline{0,Y-1},\ x=\overline{0,X-1})}$ размером $Y \times X$ пикселей формируется матрица $S = \|s(y,x)\|_{(y=\overline{0,Y-1},\ x=\overline{0,X-1})}$ сегментации такого же размера, значения элементов которой указывают на номера $n_S \in [0,N_S]$ сегментов, которым они принадлежат, где N_S – число сегментов и номер последнего сегмента. На рис. 1 приведены матрица значений пикселей полутонового изображения и соответствующая ей матрица сегментации, содержащая 6 областей.

При инициализации алгоритма выращивания областей координаты начальных пикселей роста помещаются в стеки $Y_F = \|y_F(p_F)\|_{(p_F=\overline{0,P-1})}$ и $X_F = \|x_F(p_F)\|_{(p_F=\overline{0,P-1})}$ координат смежных пикселей, где p_F – указатель стеков; P_F – максимально возможное число координат в стеках Y_F и X_F. Затем реализуется цикл обработки стеков Y_F и X_F, в котором из них извлекаются координаты (y,x) очередного обрабатываемого пикселя; соответствующему элементу матрицы сегментации $s(y,x)$ присваивается значение n_S номера сегмента; проверяется выполнение условия присоединения к текущему пикселю (y,x) смежных пикселей с координатами $(y+y', x+x')$ при $(y'=\overline{-1,1})\wedge(x'=\overline{-1,1})\wedge((y'\neq 0)\vee(x'\neq 0))$; если для какого-либо смежного пикселя условие выполняется, то его координаты $(y+y', x+x')$ заносятся в стеки Y_F и X_F. Цикл повторяется до тех пор, пока все начальные пиксели роста не будут обработаны.

Загрузка стеков максимальна, когда размер сегмента совпадает с размером изображения. В этом случае $N_S=1$, $\forall y \forall x (s(y,x) = N_S)$, размер $R_S(n_S)$ n_S-го сегмента, определяемый с помощью выражения $R_S(n_S) = \frac{1}{n_S}\sum_{y=0}^{Y-1}\sum_{x=0}^{X-1} s(y,x)$, составляет $Y \times X$

100	100	50	50	50	100	100
100	50	50	50	50	50	100
50	50	50	100	50	50	50
50	50	100	100	100	50	50
50	50	50	100	50	50	50
100	50	50	50	50	50	100
100	100	50	50	50	100	100

a

1	1	2	2	2	3	3
1	2	2	2	2	2	3
2	2	2	4	2	2	2
2	2	4	4	4	2	2
2	2	2	4	2	2	2
5	2	2	2	2	2	6
5	5	2	2	2	6	6

b

Рис. 1. Матрицы:
a – матрица значений пикселей; *b* – матрица сегментации
Fig. 1. Matrices:
a – matrix of pixel values; *b* – segmentation matrix

($R_S(1) = YX$) при исключении повторного попадания координат в стеки. При отсутствии выражения для точного определения размера стека (значения P_F числа ячеек стека) гарантировать устойчивую работу алгоритма выращивания областей, исключающую переполнение стеков Y_F и X_F, можно только если

$$P_F = YX . \quad (1)$$

Выражение (1) не учитывает выборку координат из стеков, обеспечивающую выполнение неравенства $P_F < YX$. Таким образом, для повышения эффективности использования оперативной памяти при сегментации изображений на основе выращивания областей необходимо точно определить размер P_F стеков Y_F и X_F при максимальной загрузке, когда размер единственной области $R_S(1)$ (N_S=1) совпадает с размером изображения ($R_S(1) = YX$).

Уточнение размера FIFO-стека при максимальной загрузке. Пусть квадратное изображение размером Y^2 пикселей содержит единственную область такого же размера. Работу FIFO-стека в процессе сегментации такой области можно разделить на циклы. В пределах каждого цикла из FIFO-стека выгружаются все координаты пикселей, которые там находились в начале цикла, осуществляется их обработка и загрузка координат пикселей, смежных с обработанными.

Число координат, загруженных в FIFO-стек в каждом цикле, зависит от положения начального пикселя роста (край или центр области). Если начальный пиксель роста находится в центре квадратной области, то в каждом цикле число элементов в FIFO-стеке увеличивается на 8 (рис. 2,*a*). Тогда число координат, загружаемых в FIFO-стек в последнем ((Y −1/2))-м цикле составляет 4(Y−1) при условии, что Y – нечетное. Если начальный пиксель роста находится в углу квадратной области, то в каждом цикле число координат в FIFO-стеке увеличивается на 2 (рис. 2,*b*). Тогда число координат, загружаемых в FIFO-стек в последнем (Y −*1*)-м цикле, составляет 2Y −1. Таким образом, при сегментации области с положением начального пикселя роста в ее центре, требуется стек большей емкости (примерно в 2 раза), чем в случае, когда начальный пиксель роста расположен в углу области, а также в произвольном месте на границе области (рис. 2,*c*).

Для расчета емкости стека необходимо учитывать порядок выборки значений смежных пикселей. Окрестность углового пикселя (рис. 3,*a*) может содержать до 5-ти необработанных смежных пикселей, а окрестности других (неугловых) пикселей – до трех пикселей (рис. 3,*b*). Это значит, что при выборке из FIFO-стека координат углового пикселя на их место могут вернуться координаты 5-ти смежных пикселей и число координат в FIFO-стеке увеличится на 4. При выборке из FIFO-стека координат неуглового пикселя на их место могут вернуться координаты трех смежных пикселей и число координат в FIFO-стеке увеличится на 2.

В рассматриваемом случае сегментации квадратной области с начальным пикселем роста в её центре наибольшее число координат в FIFO-стеке накапливается, если в процессе обработки окрестности области смежные пиксели выбираются не по порядку, а с интервалом два

3	3	3	3	3	3	3
3	2	2	2	2	2	3
3	2	1	1	1	2	3
3	2	1	0	1	2	3
3	2	1	1	1	2	3
3	2	2	2	2	2	3
3	3	3	3	3	3	3

a

6	6	6	6	6	6	6
5	5	5	5	5	5	6
4	4	4	4	4	5	6
3	3	3	3	4	5	6
2	2	2	3	4	5	6
1	1	2	3	4	5	6
0	1	2	3	4	5	6

b

6	6	6	6	6	6	6
5	5	5	5	5	5	5
4	4	4	4	4	4	4
3	3	3	3	3	3	4
2	2	2	2	2	3	4
2	1	1	1	2	3	4
2	1	0	1	2	3	4

c

Рис. 2. Распределение пикселей сегментируемой квадратной области по циклам обработки FIFO-стеков Y_F и X_F при различном положении начального пикселя роста: *a* – в центре области; *b* – в углу области; *c* – на краю области

Fig. 2. The distribution of pixels of the segmented square region according to the processing cycles of the FIFO stacks Y_F and X_F at different positions of the initial growth pixel:
a – in the center of the region; *b* – in the corner of the region; *c* – on the edge of the area

пикселя (рис. 3,в) и выборка начинается с 4-х угловых пикселей. Выгрузка из FIFO-стека координат 4-х угловых пикселей, выращиваемой области, приводит к возврату в FIFO-стек координат 16-ти смежных пикселей. Выгрузка из FIFO-стека координат неугловых пикселей с интервалом два пикселя в *k*-м цикле (всего $4(\lfloor y/3 \rfloor - 1)$ пикселей при $y = 2k + 1$, $k = \overline{1, (Y-1)/2}$ и нечетном значении *Y*) приводит к возврату в FIFO-стек координат $8(\lfloor y/3 \rfloor - 1)$-ти смежных пикселей, где $\lfloor\ \rfloor$ – операция округления в меньшую сторону. Выгрузка из FIFO-стека координат остальных неугловых пикселей не приводит к загрузке в стек новых координат (координаты всех смежных пикселей уже находятся в стеке) и число координат в FIFO-стеке уменьшается.

Таким образом, к концу цикла число координат в FIFO-стеке увеличится на 8, по сравнению с предыдущим циклом, однако в начале цикла разница в числе координат в FIFO-стеке между соседними циклами может достигать $16+8(\lfloor y/3 \rfloor - 1)$ при выборке смежных пикселей с интервалом два пикселя, начиная с угловых пикселей. В этом случае в последнем цикле, когда размер сегментируемой области совпадает с размером изображения, число координат в FIFO-стеке достигает максимального значения, определяющего необходимый размер P_F FIFO-стека (число ячеек), вычисляемый с помощью выражения

$$P_F = 16 + 8(\lfloor Y/3 \rfloor - 1). \qquad (2)$$

В случае прямоугольного изображения размером $Y \times X$ для определения необходимого размера FIFO-стека по выражению (2) вместо значения *Y* предлагается использовать значение $\sqrt{YX}$.

Уточнение размера LIFO-стека при максимальной загрузке. Пусть квадратное изображение размером Y^2 пикселей содержит единственный сегмент такого же размера. Число координат, загружаемых в LIFO-стек при обработке окрестности текущего пикселя, зависит от порядка выборки смежных пикселей и положения текущего обрабатываемого пикселя относительно границ сегмента.

Текущий и смежные с ним пиксели сканируются квадратным окном размером 3x3 пикселя. Возможны два базовых варианта выборки 8-ми смежных пикселей в окне сканирования: когда последние загружаемые в стеки координаты пикселя указывают на сторону (например, верхнюю (рис. 4, *a*) или угол (например, верхний правый (рис. 4, *e*) окна сканирования. Существуют всего 16 вариантов последовательной выборки смежных пикселей с учетом движения по и против часовой стрелки. Если окно сканирования не находится на границах сегмента и первый выбираемый пиксель расположен в углу окна, то выборка смежных пикселей приводит к вертикальному или горизонтальному перемещению окна сканирования (например, вверх для окна с расположением первого выбираемого пикселя в правом верхнем

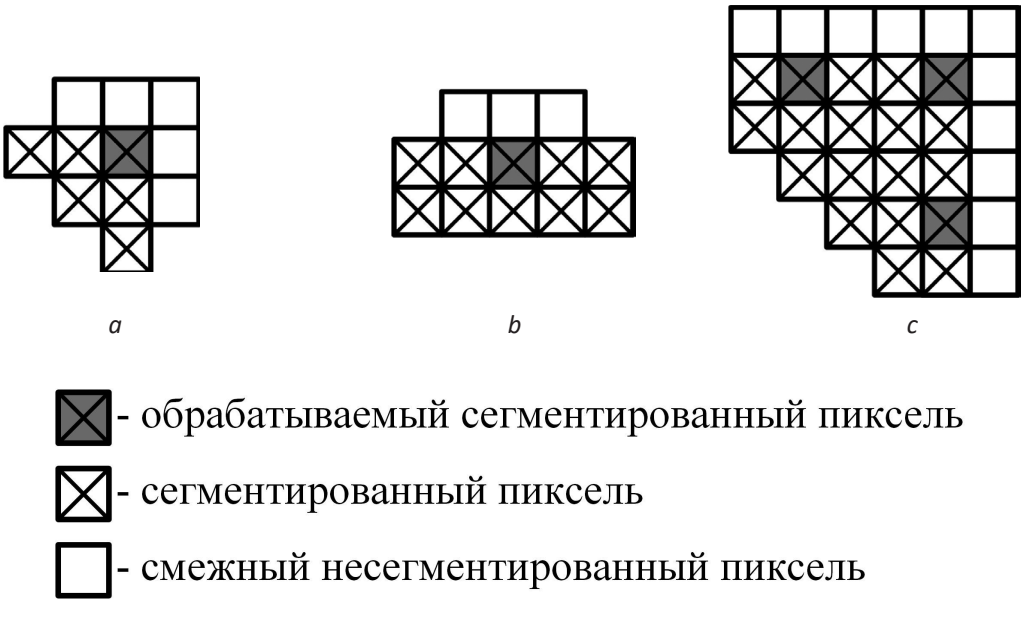

Рис. 3. Распределение пикселей сегментируемой области:
a – в окрестности углового пикселя; b – в окрестности неуглового пикселя;
c – при выборке с интервалом два пикселя
Fig. 3. The pixel distribution of the segmented area:
a – in the vicinity of the corner pixel; b – in the vicinity of a non-corner pixel;
c – when sampling with an interval of two pixels

углу (рис. 4,*b*). При расположении в углу окна сканирования последнего выбираемого пикселя выборка приводит к диагональному перемещению окна сканирования (например, вверх и вправо для окна с расположением последнего выбираемого пикселя в правом верхнем углу (рис. 4,*f*). Для каждого из 16 вариантов последовательной выборки смежных пикселей существуют Y^2 вариантов траекторий перемещения окна сканирования в пределах границ квадратного сегмента размером Y^2 пикселей в зависимости от положения начального пикселя роста сегмента (начального положения окна сканирования). На рис. 4 приведены примеры траекторий перемещения окна сканирования в пределах границ квадратного сегмента для двух вариантов выборки смежных пикселей в окне сканирования, приведенных на рис. 4,*a* и 4,*e*, когда положение начального пикселя роста соответствует нижнему левому углу (рис. 4,*c*, 4,*g*) и центру (рис. 4,*d*, 4,*h*) сегмента.

Для траектории сканирования, приведенной на рис. 4,c, необходимо $\lfloor Y/4 \rfloor$ циклов, в каждом из которых обрабатываются $4(Y–4k)$ пикселей, где $k = \overline{1,\lfloor Y/4 \rfloor}$, – номер цикла. При каждом перемещении окна из стеков извлекаются координаты текущего пикселя и помещаются в стеки координаты двух смежных пикселей, в результате чего число координат в стеках увеличивается на единицу. При обработке первого пикселя в левом нижнем углу сегмента в стеки помещаются коор-

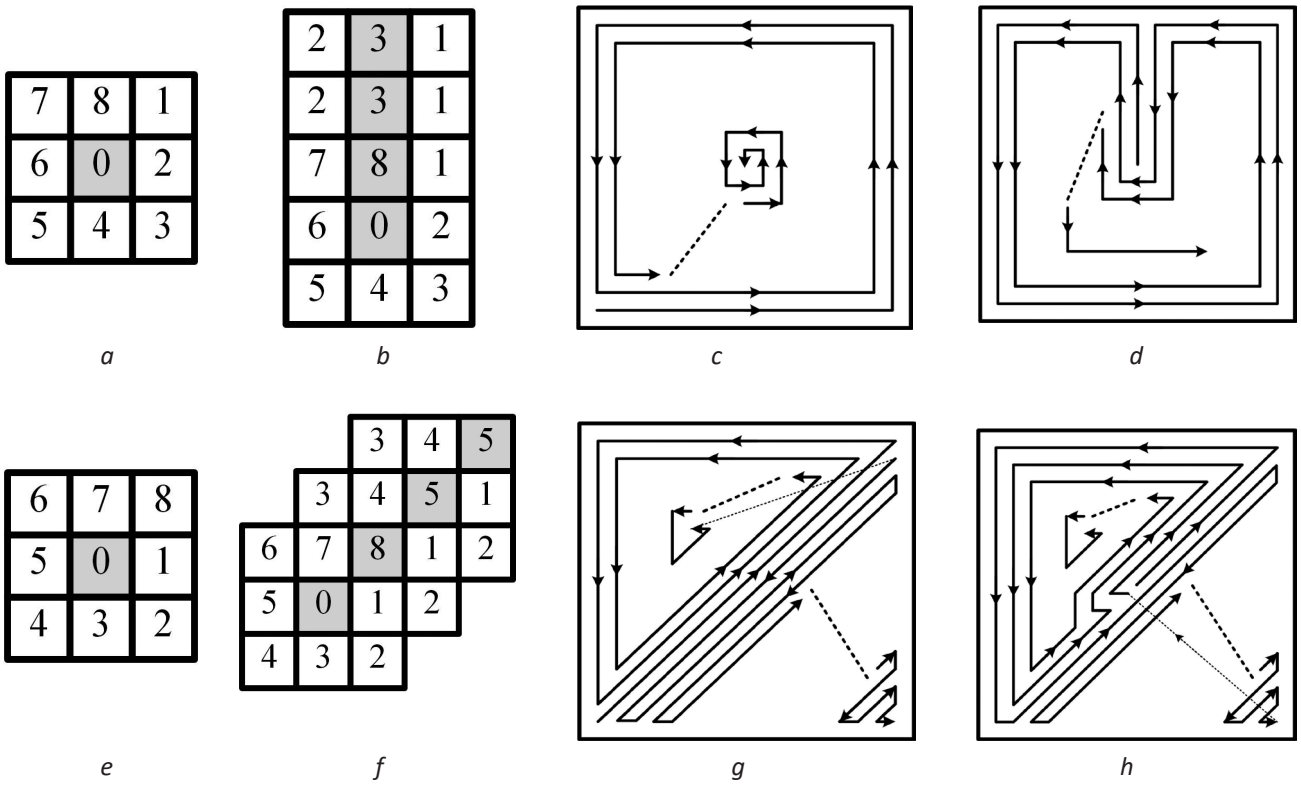

Рис. 4. Фрагменты: *a* – окно сканирования (первый выбираемый пиксель в правом верхнем углу); *b* – перемещение окна сканирования вверх; *c* – перемещение окна сканирования (первый выбираемый пиксель в правом верхнем углу окна сканирования, начальный пиксель роста в нижнем левом углу сегмента); *d* – перемещение окна сканирования (первый выбираемый пиксель в правом верхнем углу окна сканирования, начальный пиксель роста в центре сегмента); *e* – окно сканирования (последний выбираемый пиксель в правом верхнем углу); *f* – перемещение окна сканирования вверх и вправо; *g* – перемещение окна сканирования (последний выбираемый пиксель в правом верхнем углу окна сканирования, начальный пиксель роста в нижнем левом углу сегмента); *h* – перемещение окна сканирования (последний выбираемый пиксель в правом верхнем углу окна сканирования, начальный пиксель роста в центре сегмента)

Fig. 4. Fragments: *a* – scanning window (first selectable pixel in the upper right corner); *b* – moving the scan window up; *c* – moving the scanning window (the first selectable pixel in the upper right corner of the scanning window, the initial growth pixel in the lower left corner of the segment); *d* – moving the scan window (first selectable pixel in the upper right corner of the scan window, the initial growth pixel in the center of the segment); *e* – scan window (last selected pixel in the upper right corner); *f* – moving the scan window up and to the right; *g* – moving the scanning window (the last selected pixel in the upper right corner of the scanning window, the initial growth pixel in the lower left corner of the segment); *h* - moving the scanning window (the last selected pixel in the upper right corner of the scanning window, the initial growth pixel in the center of the segment)

динаты еще одного пикселя. Таким образом, размер P_F LIFO-стека для траектории сканирования, приведенной на рис. 4,*c*, определяется с помощью выражения

$$P_F = 1+4\sum_{k=1}^{\lfloor Y/4\rfloor}(Y-4k). \quad (3)$$

Для траектории сканирования, приведенной на рис. 4,*d*, необходимо $\lfloor Y/8\rfloor$ циклов, в каждом из которых обрабатываются $5(Y-8k)$ пикселей ($4(Y-8k)$ по периметру сегмента и $(Y-8k)$ в центральной части сегмента), где $k=\overline{1,\lfloor Y/8\rfloor}$. При каждом перемещении окна из стеков извлекаются координаты текущего пикселя и помещаются в стеки координаты двух смежных пикселей, в результате чего число координат в стеках увеличивается на единицу. При обработке первого пикселя в центре сегмента в стеки помещаются координаты семи пикселей, а затем, при каждом перемещении окна вверх, в стеки добавляются координаты двух пикселей (всего *Y* пикселей). При обработке пикселей в центральной части сегмента в каждом цикле в стеки помещаются координаты еще 10 пикселей. Таким образом, размер P_F LIFO-стека для траектории сканирования, приведенной на рис. 4,*d*, определяется с помощью выражения

$$P_F = 7 + Y + 5\sum_{k=1}^{\lfloor Y/8\rfloor}(Y-8k) + 10\left\lfloor\frac{Y}{8}\right\rfloor. \quad (4)$$

Траектория сканирования, приведенная на рис. 4*g*, состоит из двух фрагментов. При переходе от обработки первого фрагмента ко второму из стеков выгружается большая часть координат смежных пикселей.

Для первого фрагмента необходимо $\lfloor Y/5\rfloor$ циклов. При каждом перемещении окна вверх из стеков извлекаются координаты текущего пикселя и помещаются в стеки координаты 5-ти смежных пикселей в первом цикле (число координат в стеках увеличивается на 4) и 3-х пикселей в последующих циклах (число координат в стеках увеличивается на два). При каждом перемещении окна влево и вниз из стеков извлекаются координаты текущего пикселя и помещаются в стеки координаты двух смежных пикселей, в результате чего число координат в стеках увеличивается на единицу. При этом число координат смежных пикселей в LIFO-стеке определяется выражением

$$P_F = 4Y + 2\sum_{k=1}^{\lfloor Y/5\rfloor}(Y-5k) + 2\sum_{k=1}^{\lfloor Y/5\rfloor}(Y-4k)\ . \quad (5)$$

Для второго фрагмента необходимо $\lfloor Y/3\rfloor$ циклов. При каждом перемещении окна из стеков извлекаются координаты текущего пикселя и помещаются в стеки координаты 3-х смежных пикселей, в результате чего число координат в стеках увеличивается на два. С учетом того, что после обработки первого фрагмента в LIFO-стеке остаются координаты $4Y$ смежных пикселей, размер LIFO-стека определяется выражением

$$P_F = 4Y + 2\sum_{k=1}^{\lfloor Y/3\rfloor}(Y-3k). \quad (6)$$

Траектория сканирования, приведенная на рис. 4*h*, состоит из двух фрагментов. При переходе от обработки первого фрагмента ко второму из стеков выгружается большая часть координат смежных пикселей.

Первый фрагмент включает перемещение окна относительно центра сегмента вверх (при каждом перемещении число координат в стеках увеличивается на 4), влево (число координат в стеках увеличивается на единицу), вниз (число координат в стеках увеличивается на единицу) и еще $\lfloor Y/3\rfloor$ циклов перемещения вверх и вниз (число координат в стеках увеличивается на два). Размер LIFO-стека определяется выражением

$$P_F = 6Y + 2\sum_{k=1}^{\lfloor Y/3\rfloor}(Y-3k). \quad (7)$$

Для второго фрагмента необходимо $\lfloor Y/5\rfloor$ циклов. При каждом перемещении окна вверх из стеков извлекаются координаты текущего пикселя и помещаются в стек координаты 3-х пикселей (число координат в стеках увеличивается на два). При каждом перемещении окна влево и вниз из стеков извлекаются координаты текущего пикселя и помещаются в стеки координаты двух смежных пикселей, в результате чего число координат в стеках увеличивается на единицу. С учетом того, что после обработки первого фрагмента в LIFO-стеке остаются координаты $6Y$ смежных пикселей размер LIFO-стека определяется выражением

$$P_F = 6Y + 2\sum_{k=1}^{\lfloor Y/5\rfloor}(Y-5k) + 2\sum_{k=1}^{\lfloor Y/5\rfloor}(Y-4k)\ . \quad (8)$$

В случае прямоугольного изображения размером $Y\times X$ для определения необходимого размера LIFO-стека по выражениям (3) – (8) вместо значения *Y* предлагается использовать значение $\lceil\sqrt{YX}\rceil$, где $\lceil\ \rceil$ – операция округления в большую сторону.

Оценка экономии памяти при уточнении размеров FIFO- и LIFO-стеков. На рис. 5 приведены за-

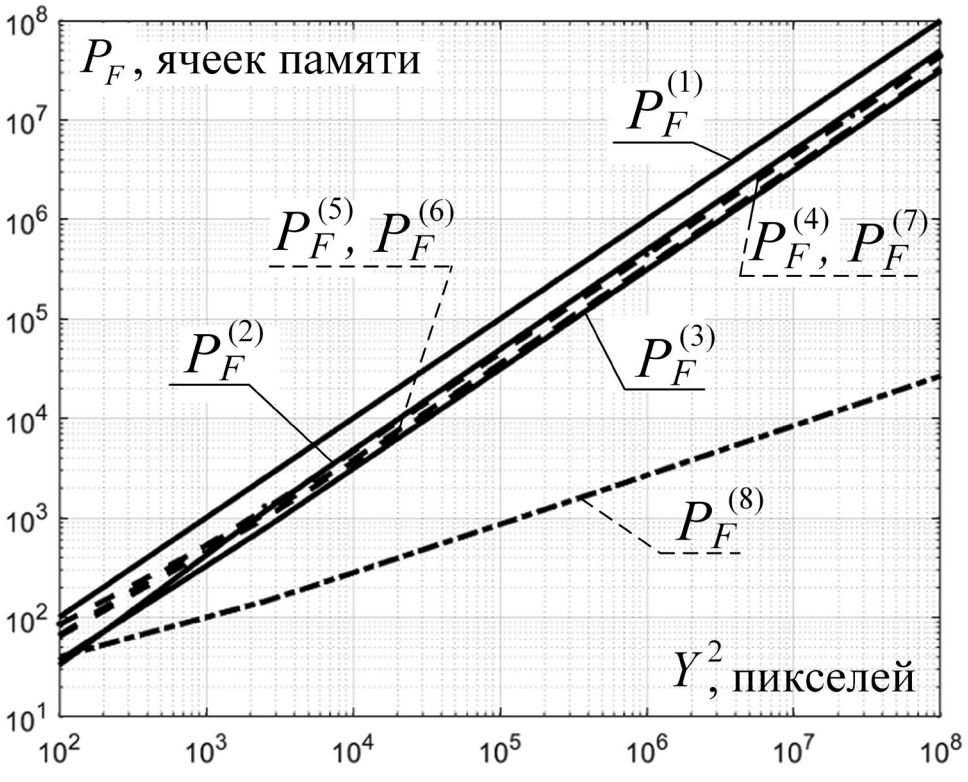

Рис. 5. Зависимости размеров FIFO- и LIFO-стеков от размера изображения
Fig. 5. Dependencies of FIFO and LIFO stack sizes on image size

висимости размеров $P_F^{(E)}$ (число ячеек памяти) FIFO- и LIFO-стеков, определяемых с помощью выражений (1) – (8), от размера Y^2 (число пикселей) квадратного изображения, где E – номер выражения.

Из рис. 5 следует, что наибольшие значения размера LIFO-стека дает выражение (3), что соответствует перемещению окна сканирования в пределах сегмента по траектории, приведенной на рис. 4с. При этом наибольшие значения размера FIFO-стека, определяемые по выражению (2), значительно меньше. Таким образом, использование FIFO-стека для сегментации изображений на основе выращивания областей предпочтительнее, чем LIFO-стека.

Экономия M_E памяти при точном расчете необходимого размера FIFO-стека по выражению (2), в сравнении с приблизительным расчетом по выражению (1), определяется с помощью отношения

$$M_E = \frac{YX}{16 + 8(\lfloor\sqrt{YX}/3\rfloor - 1)} \ . \qquad (9)$$

Из рис. 5 следует, что экономия памяти при точном расчете необходимого размера FIFO-стека растет с увеличением размера изображения и составляет 374 раза для изображения 1000×1000 пикселей.

Заключение. Для алгоритма сегментации на основе выращивания областей получены выражения, позволяющие повысить точность определения размеров FIFO- и LIFO-стеков, необходимых для хранения координат смежных пикселей. При этом учтены условия максимальной загрузки стеков, когда: а) осуществляется сегментация квадратной области с начальным пикселем роста в центре (для FIFO) и в углу (для LIFO) этой области; б) в процессе обработки окрестности области смежные пиксели выбираются не по порядку с интервалом два пикселя, начиная с 4-х угловых пикселей (для FIFO), и по порядку с расположением первого выбираемого пикселя в углу окна сканирования (для LIFO). Сравнительная оценка размеров стеков, необходимых для сегментации изображений на основе выращивания областей, показала, что использование FIFO-стека предпочтительнее, чем LIFO-стека. При этом экономия памяти составляет 374 раза для изображения 1000×1000 пикселей.

Список литературы

1. Otsu, N. A threshold selection method from gray-level histograms / N. Otsu // IEEE transactions on systems, man, and cybernetics. – 1979. – Vol. 9. – Pp. 62–66.
2. Haralick, R.M. Image segmentation techniques / R.M. Haralick, L.G. Shapiro // Computer Vision, Graphics, and Image Processing. – 1985. – Vol. 29 (1). – Pp. 100–132.
3. Horowitz, S.L. Picture segmentation by a tree traversal algorithm / S.L. Horowitz, T. Pavilidis // Journal of the ACM. – 1976. – Vol. 23 (2). – Pp. 368–388.
4. Meyer, F. Topographic distance and watershed lines / F. Meyer // Signal Processing. – 1994. – Vol. 38 (1). – P. 113–125.
5. Jain, P.K. An adaptive single seed based region growing algorithm for color image segmentation / P.K. Jain, S. Susan // India Conference (INDICON) 2013. Annual IEEE. – 2013. – Pp. 1–6.
6. A Region Growing Vessel Segmentation Algorithm Based on Spectrum Information / H. Jiang [et al.] // Computational and Mathematical Methods in Medicine: Hindawi Publishing Corporation. – 2013. – Pp. 1–9.
7. Merzougui, M. Region growing segmentation optimized by evolutionary approach and Maximum Entropy / M. Merzougui, A.E. Allaoui // International Workshop on Microwave Engineering, Communications Systems and Technologies (MECST'2019), Leuven, Belgium. – 2019. – P. 1046–1051.
8. Brass, P. Advanced Data Structures / P. Brass. – New York: Cambridge University Press, 2008. – 456 p.
9. Fox, C. Concise Notes on Data Structures and Algorithms / C. Fox. – James Madison University, 2011. – 136 p.
10. Goodrich, M.T. Data Structures and Algorithms in Java / M.T. Goodrich, R. Tamassia, M.H. Goldwasser. – Wiley, 2014. – 736 p.

References

1. Otsu N. A threshold selection method from gray-level histograms. IEEE transactions on systems, man, and cybernetics, 1979, vol. 9, pp. 62–66.
2. Haralick R. M., Shapiro L. G. Image segmentation techniques. Computer Vision, Graphics, and Image Processing, 1985, vol. 29 (1), pp. 100–132.
3. Horowitz S.L., Pavilidis T. Picture segmentation by a tree traversal algorithm. Journal of the ACM, 1976, vol. 23 (2), pp. 368–388.
4. Meyer F. Topographic distance and watershed lines. Signal Processing, 1994, vol. 38 (1), pp. 113–125.
5. Jain P. K., Susan S. An adaptive single seed based region growing algorithm for color image segmentation. India Conference (INDICON) 2013. Annual IEEE, pp. 1–6.
6. Jiang H. [et al.]. A Region Growing Vessel Segmentation Algorithm Based on Spectrum Information. Computational and Mathematical Methods in Medicine: Hindawi Publishing Corporation, 2013, pp. 1–9.
7. Merzougui M., Allaoui A.E. Region growing segmentation optimized by evolutionary approach and Maximum Entropy. International Workshop on Microwave Engineering, Communications Systems and Technologies (MECST'2019), Leuven, Belgium, 2019, pp. 1046–1051.
8. Brass P. Advanced Data Structures. New York: Cambridge University Press, 2008. 456 p.
9. Fox C. Concise Notes on Data Structures and Algorithms. James Madison University, 2011. 136 p.
10. Goodrich M. T., Tamassia R., Goldwasser M. H. Data Structures and Algorithms in Java. Wiley, 2014. 736 p.

Received: 05.03.2020 *Поступила: 05.03.2020*