Математическое и компьютерное моделирование:
материалы III Международной научной конференции
(Омск, 12 ноября 2015 г.). Омск, 2015. С. 153–155. УДК 004.92

П.Н. Лазаренко, Д.Н. Лавров

Омский государственный университет им. Ф.М. Достоевского, г. Омск

ОЦЕНКА АЛГОРИТМОВ СКЕЛЕТИЗАЦИИ В РАМКАХ ИССЛЕДОВАНИЙ ПО СЕГМЕНТАЦИИ КРОВЕНОСНЫХ СОСУДОВ

Данные исследования проводились в рамках проектирования и разработки СУД основанной на идентификации рисунка вен пальца. Целью данной работы является определение оптимального алгоритма скелетизации, который будет использоваться для дальнейшей сегментации вен.

1. Обзор исследуемых алгоритмов скелетизации

8-ядерный алгоритм. Скелетизация проводится с помощью ядер изображённых на рис. 1. Алгоритм итеративный и состоит из следующих шагов (для каждого пикселя) [1, с. 22–24]:

1) создаётся матрица с белыми пикселями равными 1, в этом случае чёрные пиксели равны 0;

2) создаётся матрица с белыми пикселями равными 0, и чёрными равными 1;

3) каждое ядро применяется к обеим матрицам;

4) каждое ядро создаёт матрицу в качестве результата для обеих матриц;.

5) применить двоичный порог равный 2,99 для обеих матриц;

6) удалить пиксель из источника, если обе матрицы являются правдой;

7) продолжать процесс до тех пор, пока ни один пиксель не будет удалён.

После скелетизации изображение будет иметь различные ненужные артефакты, такие как мелкие ветви и отверстия. Для их устранения после скелетизации изображение размывается фильтром Гаусса и процедура повторяется.

Алгоритм Зонга–Суня. Основная идея алгоритма Зонга–Суня заключается в том, что на каждом шаге, пробегая по изображению рамкой 3x3, проверяется принадлежность каждого пикселя к гра-

© Лазаренко П.Н., Лавров Д.Н., 2015

нице заданной связной области. Если условия проверки выполняются, то пиксель удаляется из области. Вне зависимости от количества выполненных шагов область останется связной, в предельном случае она выродится в линию толщиной в один пиксель.

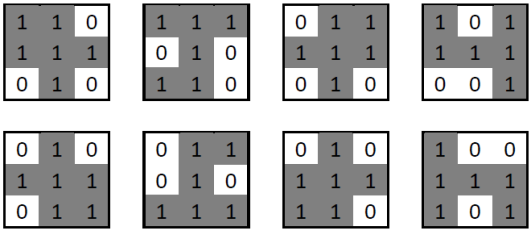

Рис. 1: Ядра, используемые в 8-ядерном алгоритме [1]

1. **While** points are deleted **do**
2. **For all** pixels $p(i,j)$ **do**
3. **if** (a) $2 \le B(P_1) \le 6$
   (b) $A(P_1) = 1$
   (c) Apply one of the following:
   1. $P_2 \times P_4 \times P_6 = 0$ in odd iterations
   2. $P_2 \times P_4 \times P_8 = 0$ in even iterations
   (d) Apply one of the following:
   1. $P_4 \times P_6 \times P_8 = 0$ in odd iterations
   2. $P_2 \times P_6 \times P_8 = 0$ in even iterations
   then
4. Delete pixel $p(i,j)$
5. **end if**
6. **end for**
7. **end while**

Для каждого пикселя:
P9 P2 P3
P8 P1 P4
P7 P6 P5

Рис. 2: Алгоритм Зонга–Суня [2]

Алгоритм Зонга–Суня представлен на рис. 2, где $A(P_1)$ – число переходов от 0 к 1 в окрестности пиксела, т.е. в последовательности P_2, P_3, P_4, P_5, P_6, P_7, P_8, P_9, P_2. $B(P_1)$ – число ненулевых соседей пикселя. Таким образом получается следующий алгоритм.

Подытерация 1 (c). Удаление точек на юго-восточной границе и северо-западных угловых точек.

Подытерация 2 (d). Удаление точек на северо-западной границе и юго-восточных угловых точек.

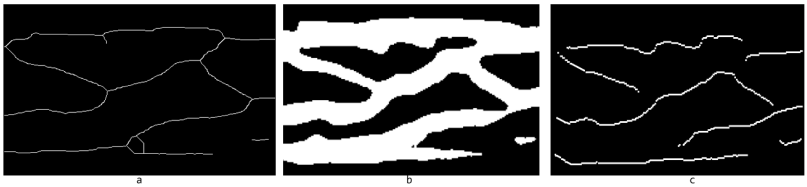

Рис. 3: Результат работы алгоритмов: (a)Зонг–Сунь, (b)Исходное изображение, (c)8-ядерный алгоритм

2. Сравнение реализованных алгоритмов

Эффективность. Для проверки качества работы алгоритмов использовался метод экспертной оценки. Группе из 10 человек были показаны результаты работы алгоритмов. Единогласно предпочтения были отданы алгоритму Зонга–Суня. Итог работы алгоритмов изображён на рисунке 3.

Скорость работы. Данные алгоритмы были реализованы на языках Java и Scilab. Результаты тестов производительности алгоритмов на одном и том же изображении приведены в таблице 1.

Таблица 1: Результаты измерения производительности

Алгоритм	Java	Scilab
Зонга–Суня	7.83 мс	67с
8-ядерный алгоритм	4.7 мс	40с

Первый вывод: реализация алгоритмов в Scilab из-за низкой производительности подходит только для быстрого прототипирования. Второй вывод: скорость 8-ядерного алгоритма выше, но незначительно, чтобы жертвовать качеством. Более точная оценка влияния алгоритмов на конечную цель создания системы (идентификацию пользователя по рисунку вен) может быть произведена лишь в полной системе, содержащей все модули обработки данных.

Литература

1. *Vehils D., Miguel J.* Design and Implementation of a Finger Vein Identification System. Polytechnic University of Catalania, 2011.

2. Implementation of thinning algorithm in OpenCV. URL: `http://opencv-code.com/quick-tips/implementation-of-thinning-algorithm-in-opencv`.