Оригинальная статья

https://doi.org/10.21869/2223-1560-2023-27-1-103-113

Рекурсивный алгоритм закрашивания областей распознанных объектов

М. В. Бобырь [1] ✉, Н. И. Храпова [1], О. Г. Супрунова [1], А. А. Дородных [1]

[1] Юго-Западный государственный университет
ул. 50 лет Октября, д. 94, г. Курск 305040, Российская Федерация

✉ e-mail: maxbobyr@gmail.com

Резюме

Цель исследования. *Разработка рекурсивного алгоритма закрашивания распознанных областей на бинаризованном изображении и выделение контуров цветовой метки с целью нахождения минимального времени подсчета количества закрашенных пикселей в распознанной метке.*

Методы. *Алгоритм закрашивания областей основывается на рекурсивном методе. Алгоритм начинает работу с центральной точки изображения и анализирует наличие незакрашенных пикселей в смежных ячейках. Рассматривается восемь направлений обхода пикселей относительно начальной точки с целью определения наименьшего времени закрашивания распознанной области. Обход осуществляется в следующих направлениях: восток-юг-запад-север; восток-север-запад-юг; юг-восток-север-запад; юг-запад-север-восток; запад-юг-восток-север; запад-север-восток-юг; север-восток-юг-запад; север-запад-юг-восток. Алгоритм содержит несколько этапов: проверка выхода из рекурсии при условии закрашивания всех областей, закрашивание исходной ячейки, закрашивание элементов по четырем направлениям от начальной точки, подсчет количества закрашенных элементов.*

Результаты. *Разработан рекурсивный алгоритм закрашивания распознанных областей на бинаризованном изображении, с возможностью выделения контуров распознанной метки. Определено направление оптимального обхода, который имеет наименьшее время по отношению к другим рассматриваемым направлениям. Тестирование осуществлялось для циклов с 10, 50 и 100 итерациями. По представленному алгоритму создана специализированная программная модель. Номер свидетельства о государственной регистрации программы для ЭВМ «Программа для заливки распознанной метки» – 2023612631.*

Заключение. *Результаты экспериментальных исследований показали, что для значения цикла в 10 итераций наилучшее время закрашивания области равно 12762 мс, для значения цикла в 50 итераций наилучшее время равно 76008 мс, для значения цикла в 100 итераций наилучшее время равно160568 мс. Минимальное среднее время выполнения операции по закрашиванию составило 84357 мс, следовательно, наилучшей из восьми комбинаций прохода оказалось направление – север-восток-юг-запад.*

Ключевые слова: *рекурсивный алгоритм; закрашивание областей; распознавание метки; направления обхода; быстродействие.*

Конфликт интересов: *Авторы декларируют отсутствие явных и потенциальных конфликтов интересов, связанных с публикацией настоящей статьи.*

Финансирование: *Работа выполнена при поддержке гранта РНФ 23-21-00071 – «Разработка модели компьютерного зрения для интеллектуальной навигации робототехнических систем, основанной на построении трехмерных сцен по картам глубин».*

Для цитирования: Рекурсивный алгоритм закрашивания областей распознанных объектов / М. В. Бобырь, Н. И. Храпова, О. Г. Супрунова, А. А. Дородных // Известия Юго-Западного государственного университета. 2023; 27(1): 103-113. https://doi.org/10.21869/2223-1560-2023-27-1-103-113.

Поступила в редакцию 26.01.2023 *Подписана в печать 15.02.2023* *Опубликована 14.04.2023*

***

Введение

Закрашивание – это процедура заливки необходимых областей на изображении [1]. Данная процедура используется при подсчете площади 3d объектов на картах глубин [2,3], построенных на основе нечеткой логики [4-6], в медицине, с целью выявления патологий на медицинских снимках, таких как рентгенография, флюорография, ультразвуковое исследование, магнитно-резонансная томография [7-9].

Существует несколько способов заливки областей. В работе [10] представлен алгоритм закрашивания невидимых областей, который используется в биометрических системах персональной идентификации по изображениям лиц. Для автоматической сегментации области лица предлагается метод минимального разреза графа с использованием особых точек для предварительной разметки [11-13]. В качестве таких точек используются глаза, нос и т.д. В работе [14] рассматривается алгоритм маркировки связанных областей для оценки качества семян пшеницы. Данный алгоритм позволяет присваивать уникальные метки для отдельных не пересекающихся связанных областей на изображении.

Также для выделения областей фигуры на изображении используются алгоритмы поиска в ширину и глубину [15-18]. Ограничением данных моделей является высокая чувствительность самого алгоритма к размеру изображения и количеству цветов, что может послужить причиной получения неоптимального решения, которое проявляется в том, что не все пиксели в распознанной области будут закрашены. В работе [19] для решения аналогичных задач, представлен алгоритм закрашивания областей фигуры на основе деревьев. Данный алгоритм аналогичен предыдущим методам поиска решений, отличием которого является использование дерева вместо графа, а выбор наилучшего способа заполнения изображения происходит с помощью алгоритма поиска в глубину [18]. В работе [20, 21] рассматривается алгоритм закрашивания областей на основе эвристик. Данный алгоритм основывается на применении простых правил для поиска оптимального решения задачи. Алгоритм эвристик использует две функции для оценки пикселей: первая функция оценивает пиксель по его соседям, а вторая функция оценивает пиксель по правилам закрашивания. На основе этих функций возможно оценить каждый пиксель и выбрать наилучший из них. Ограничением данной работы является затрата вычислительного времени, необходимого для проверки усло-

вий двух эвристик. В связи с этим возникает актуальная задача, заключающаяся в разработке новых моделей и методов, позволяющих закрашивать выделенные области на изображении и подсчитывать количество пикселей в них.

Материалы и методы

Предложенный нами алгоритм закрашивания областей основывается на рекурсивном методе. Данный алгоритм содержит 3 входных переменных: начальный элемент, базовый цвет и цвет заливки. Рассматривается восемь комбинаций прохода (восток-юг-запад-север; восток-север-запад-юг; юг-восток-север-запад; юг-запад-север-восток; запад-юг-восток-север; запад-север-восток-юг; север-восток-юг-запад; север-запад-юг-восток).

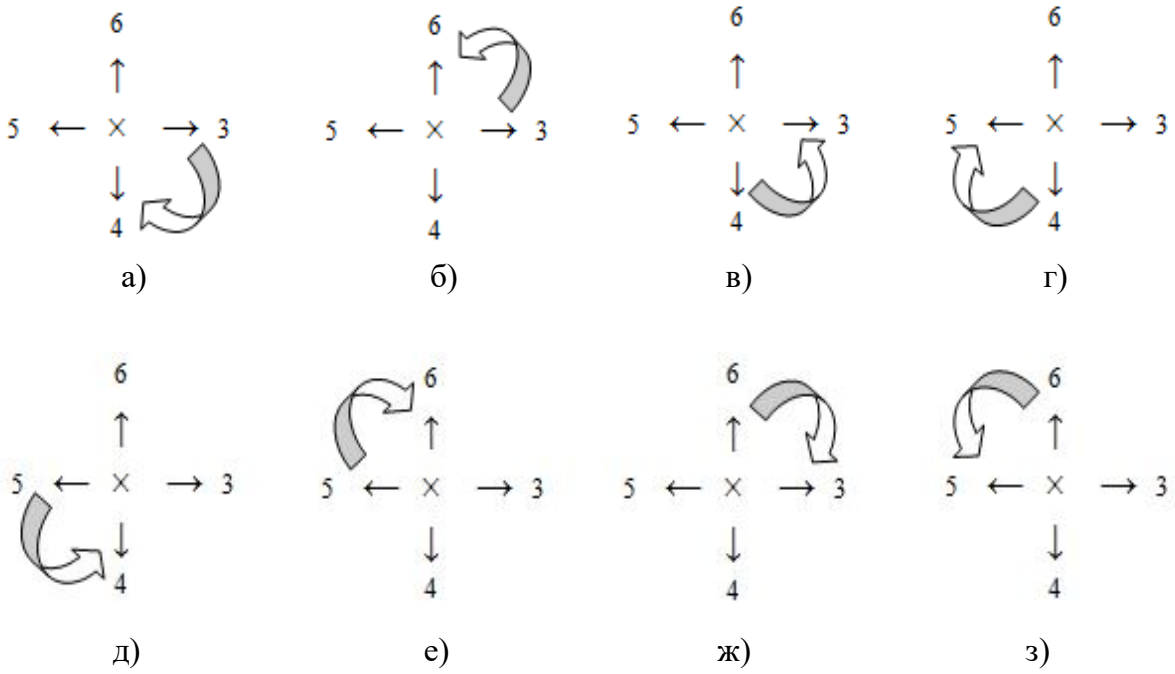

Рис. 1. Комбинации направлений обхода: **а –** восток-юг-запад-север; **б –** восток-север-запад-юг; **в** – юг-восток-север-запад; **г –** юг-запад-север-восток; **д –** запад-юг-восток-север; **е –** запад-север-восток-юг; **ж –** север-восток-юг-запад; **з –** север-запад-юг-восток

Алгоритм содержит следующие шаги:

1) проверка условия выхода из рекурсии;

2) поиск всех элементов массива;

3) определения элементов для закрашивания;

4) заливка выбранной области.

Учитывая условия для закрашивания, поочередно применяется рекурсивный вызов функции для смежных точек. Если точка не является граничной, она закрашивается.

Разработанный рекурсивный алгоритм закрашивания областей (рис.1) используется для подсчета количества пикселей и вычисления времени их закрашивания в распознанной метки.

По представленному алгоритму создана специализированная программная модель [22].

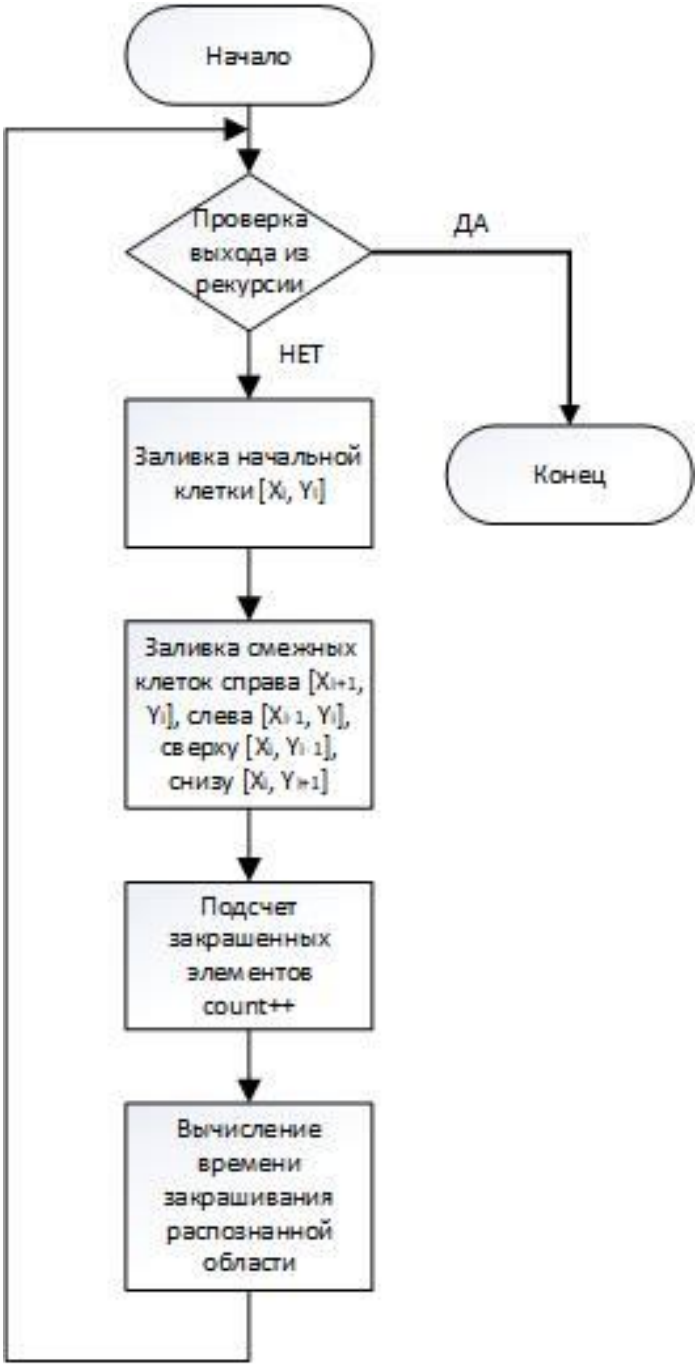

Рис. 2. Алгоритм закрашивания областей на основе рекурсии

Программа имеет несколько входных переменных: размер окна, массив для закрашивания, направление закрашивания, количество циклов закрашивания, координаты начальной точки, номер закрашиваемого элемента в массиве и граничные пиксели, которые не нужно закрашивать. Выходными переменными являются время закрашивания окна и количество элементов в закрашиваемой области.

Ниже представлен псевдокод для рекурсивного алгоритма закрашивания областей распознанных объектов.

```
Procedure Filling
Input:    int maxx, maxy – размер окна;
int N – направление закрашивания;
int n – количество циклов закрашивания;
int x, y – координаты начальной точки;
string symbols = { " ", "#", "*", ">", "v", "<", "^"} – элементы, выводимые в консоль программы;
int z – номер закрашиваемого элемента в массиве symbols;
int map[maxx, maxy] – массив для закрашивания;
Random rnd – случайное значение граничного пикселя.
Output: time – время закрашивания окна [maxx, maxy]
count = 0 – счетчик количества элементов в закрашиваемой области;
Begin
Init();                    // Инициализация процедуры
x = maxx/2; y = maxy/2;    // Установка позиции курсора в центр окна
Show(x, y, z);             // Закрашивание центральной ячейки окна
Fill(x,y);                 // Вызов функции закраски области
End
Function Init()
End
  for i = 0; i < n; i++
Show(rnd.Next(0, maxx), rnd.Next(0, maxy), 1);          // Заполнения массива окна
                                                    элементом # в случайных ячейках
Begin
Function Show(int x, int y, int z)
Begin
  map[x, y] = z;        // заполнение окна [maxx, maxy] элементами массива symbols
End
Function Fill(int x, int y)
End
  //Thread.Sleep(1);              // Создание анимации
if (map[x, y] > 0)                // Проверка условия выхода из рекурсии
                        return;   // если окно закрашено полностью – Выход

Show(x, y, 3); Fill(x + 1, y);                    // Установка направления
                                                  и закрашивание правой ячейки     ★
                 Show(x, y, 4); Fill(x, y + 1);   // Установка направления
                                                  и закрашивание нижней ячейки     ★
                 Show(x, y, 5); Fill(x – 1, y);   // Установка направления
                                                  и закрашивание левой ячейки      ★
Show(x, y, 6); Fill(x, y – 1);                    // Установка направления
                                                  и закрашивание верхней ячейки    ★
                 Show(x, y, 2);             // Закрашивание ячейки элементом «*»
                 count++;                   // Подсчет закрашиваемых элементов
Begin
```

Представленный программный код отображает комбинацию обхода пикселей, изображенную на рис. 1а (восток-юг-запад-север). Знаком «★» отмечены строки кода, которые отвечают за направления обхода относительно начальной точки. Чтобы изменить направления в программном коде необходимо поменять местами данные строки. Например, чтобы установить направление, представленное на рис. 1б (восток-север-запад-юг), программный код должен выглядеть следующим образом:

```
Begin
Function Show(int x, int y, int z)
Begin
  map[x, y] = z;    // заполнение окна [maxx, maxy] элементами массива symbols
End

Function Fill(int x, int y)
End
  //Thread.Sleep(1);              // Создание анимации
if (map[x, y] > 0)                // Проверка условия выхода из рекурсии
                      return;     // если окно закрашено полностью – выход

  Show(x, y, 3); Fill(x + 1, y);      // Установка направления и закрашивание правой ячейки
  Show(x, y, 6); Fill(x, y - 1);     // Установка направления и закрашивание верхней ячейки
  Show(x, y, 5); Fill(x - 1, y);      // Установка направления и закрашивание левой ячейки
            Show(x, y, 4); Fill(x, y + 1);   // Установка направления и закрашивание нижней ячейки

                  Show(x, y, 2);          // Закрашивание ячейки элементом «*»
                  count++;                // Подсчет закрашиваемых элементов
Begin
```

Если раскомментировать строку `//Thread.Sleep(1);`, то можно наблюдать, как программа будет проходить каждую ячейку, заполняя массив окна элементами, т.е. активируется анимационный режим.

Результаты и их обсуждение

Для экспериментальных исследований использовалось специальное программное обеспечение, разработанное авторским коллективом [22]. Программное обеспечение было разработано в среде Visual Studio 2019. Для тестирования использовался ПК со следующими характеристиками: Процессор: AMD Ryzen 5 2400G with Radeon Vega Graphics 3,60 GHz. ОЗУ: 8,00 GB.

Проведены экспериментальные исследования, в которых менялось направление закрашивания областей при значениях циклов в 10, 50 и 100 итераций.

В табл. 1 представлено полученное время выполнения операции по закрашиванию областей распознанных объектов для каждого направления обхода (см. рис.1).

Таблица 1. Время выполнения операции по закрашиванию областей распознанных объектов

Table 1. The operation execution time for filling areas of recognized objects

Направления обхода	Циклы	Время, мс	Среднее значение, мс
3 4 5 6	10	12937	84500
	50	79997	
	100	160568	
3 6 5 4	10	12762	84789
	50	79846	
	100	161760	
4 3 6 5	10	13843	85551
	50	79723	
	100	163089	
4 5 6 3	10	13417	85223
	50	80250	
	100	162003	
5 4 3 6	10	13027	85038
	50	80131	
	100	161957	
5 6 3 4	10	13561	85397
	50	80094	
	100	162537	
6 3 4 5	10	13059	84357
	50	76008	
	100	164004	
6 5 4 3	10	13185	86042
	50	82014	
	100	162927	

При сравнении полученных значений по циклам были получены следующие результаты (рис. 3):

– для значения цикла в 10 итераций наилучшее время равное 12762 мс было получено для направления восток-север-запад-юг;

– для значения цикла в 50 итераций наилучшее время равное 76008 мс было получено для направления север-восток-юг-запад;

– для значения цикла в 100 итераций наилучшее время равное 160568 мс

было получено для направления восток-юг-запад-север.

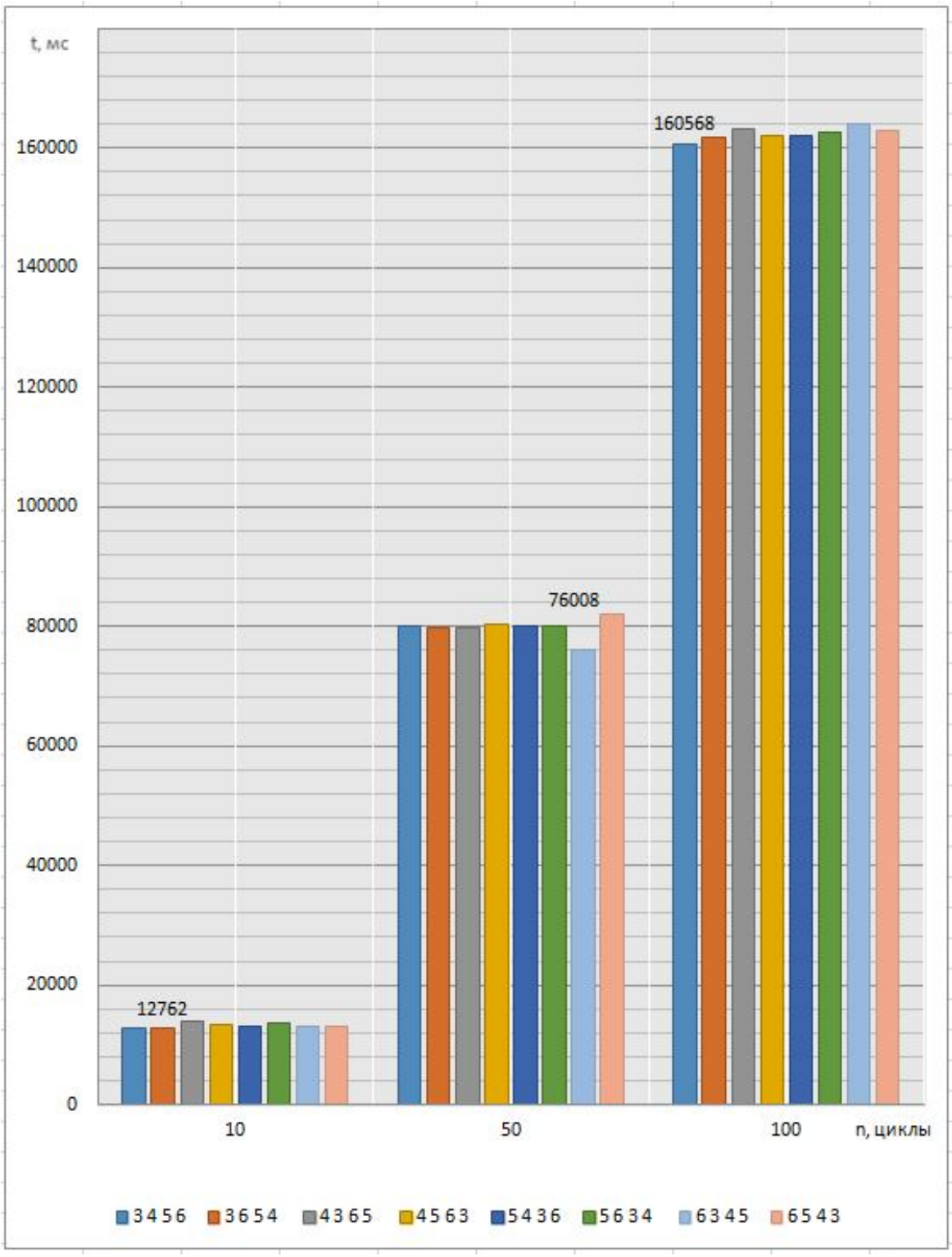

Рис. 3. Результаты экспериментальных данных

Наилучшие результаты выделены в табл. 1 полужирным начертанием.

Расчет среднего времени выполнения операции по закрашиванию для каждого направления показал следую-

щее – наилучшим направлением закрашивания областей оказалось 6-3-4-5 – север-восток-юг-запад (см. табл.1), так как при этой комбинации получено минимальное среднее время выполнения операции по закрашиванию, которое составило 84357 мс.

Выводы

В представленной работе описан рекурсивный алгоритм закрашивания распознанных областей на бинаризованном изображении, с возможностью выделения контуров распознанной метки. Проведены экспериментальные исследования, в результате которых было определено направление оптимального обхода, который имеет наименьшее время закрашивания области и подсчета количества пикселей в распознанной метке.

Список литературы

1. Бобырь М. В., Архипов А. Е., Якушев А. С. Распознавание оттенка цветовой метки на основе нечёткой кластеризации // Информатика и автоматизация. 2021. № 20(2). С. 407-434.

2. Аврашков П. П., Коськин Н. А., Константинов И. С. Оценка эффективности различных моделей конвертации изображений в стереоформат с помощью карт глубины // Научно-технический вестник Поволжья. 2020. № 12. С. 165-168.

3. Гайворонский В. А. Подход к созданию объемного панорамного изображения на основе пассивных методов определения карт глубины // Информационные системы и технологии. 2022. № 1(129). С. 24-29.

4. Бобырь М. В., Храпова Н. И., Ламонов М. А. Система управления интеллектуальным светофором на основе нечеткой логики // Известия Юго-Западного государственного университета. 2021; 25(4): 162-176. https://doi.org/10.21869/2223-1560-2021-25-4-162-176

5. Bobyr M. V., Emelyanov S. G. A nonlinear method of learning neuro-fuzzy models for dynamic control systems. 2020. Vol. 88. P. 106030. DOI 10.1016/j.asoc.2019.106030.

6. Bobyr M. V., Yakushev A. S., Dorodnykh A. A. Fuzzy devices for cooling the cutting tool of the CNC machine implemented on FPGA // Measurement. 2020. Vol. 152. P. 107378. DOI 10.1016/j.measurement.2019.107378.

7. Обнаружение опухоли мозга на основе МРТ с применением метода нечеткой кластеризации с-средних / А. Г. Зотин, Ю. А. Хамад, С. В. Кириллова [и др.] // Медицина и высокие технологии. 2018. № 1. С. 20-28.

8. Смольникова У. А., Ушков А. Д. Опыт оценки цифровых рентгенограмм с применением системы автоматического распознавания изменений в легких // Лучевая диагностика и терапия. 2020. № S1. С. 84.

9. Улучшение сегментации патологий легких и плеврального выпота на КТ-снимках пациентов с Covid-19 / Д. С. Лащенова, А. М. Громов, А. С. Конушин, А. М. Мещерякова // Программирование. 2021. № 4. С. 56-62.

10. Алгоритмы предобработки и постобработки данных для биометрических систем распознавания лиц / В. С. Горбацевич, Ю. В. Визильтер, С. Ю. Желтов, С. А. Хаин // Вестник компьютерных и информационных технологий. 2014. № 6(120). С. 9-14.

11. Мельников Б. Ф., Чурикова Н. П. Алгоритмы сравнительного исследования двух инвариантов графа // Современные информационные технологии и ИТ-образование. 2019. Т.15, №1. С. 45-51.

12. Хачумов М. В., Талалаев А. А., Хачумов В. М. Об одном эвристическом критерии в задаче определения изоморфизма графов на основе инвариантов // Современные наукоемкие технологии. 2022. № 2. С. 159-163.

13. Галкин В. А., Арьков К. А. Автоматизированная система визуализации алгоритма Дейкстры на графах // Естественные и технические науки. 2019. № 6(132). С. 190-192.

14. Ковалев А. В., Бакуменко А. Н. Алгоритм маркировки связанных областей при потоковой обработке изображения // Инженерный вестник Дона. 2022. № 4(88). С. 191-201. EDN NTTPYR.

15. Емельянов С. Г., Бобырь М. В., Крюков А. Г. Исследование свойств алгоритма поиска в ширину для нахождения маршрута передвижения роботов // Известия Юго-Западного государственного университета. 2022; 26(4): 39-56. https://doi.org/10.21869/2223-1560-2022-26-4-39-56.

16. Метод распределения нагрузки в GPU-реализации алгоритма поиска в ширину на графе / М. А. Черноскутов, Д. Г. Ермаков, М. Л. Гольдштейн, Д. А. Усталов // Научно-технический вестник Поволжья. 2014. № 2. С. 229-232. EDN SAYCFZ.

17. Зуенко А. А. Интеграция методов поиска в ширину и логического вывода для удовлетворения табличных ограничений // Онтология проектирования. 2021. Т. 11, № 4(42). С. 521-532. DOI 10.18287/2223-9537-2021-11-4-521-532.

18. Компьютерная реализация поиска в глубину / Е. А. Пронина, А. А. Лебединская, П. Г. Шурхаленко // Мировая наука. 2018. № 11(20). С. 234-237.

19. Будникова И. К., Бабкин Т. А. Интеллектуальный анализ данных на основе инструментария алгоритма C&RT (Общие деревья) // Информационные технологии в строительных, социальных и экономических системах. 2021. № 1(23). С. 112-115.

20. Мельников Б. Ф., Дудников В. А. О задаче псевдооптимального размещения графа на плоскости и эвристиках для её решения // Информатизация и связь. 2018. № 1. С. 63-70.

21. Alzakki H. M., Tsvetkov V. Yu. Selection texture regions on the image based on classification assessment density of contour elements // Big Data and Advanced Analytics. 2017. No. 3. P. 113-118.

22. Свидетельство о государственной регистрации программы для ЭВМ № 2023612631 Российская Федерация. Программа для заливки распознанной метки: № 2023610839: заявл. 23.01.2023: опубл. 06.02.2023 / М. В. Бобырь, Н. И. Храпова,

О. Г. Супрунова; заявитель Федеральное государственное бюджетное образовательное учреждение высшего образования "Юго-Западный государственный университет".

Информация об авторах

Бобырь Максим Владимирович, доктор технических наук, профессор кафедры вычислительной техники, Юго-Западный государственный университет,
г. Курск, Российская Федерация,
e-mail: fregat_mn@rambler.ru,
ORCID: http://orcid.org/0000-0002-5400-6817,
Researcher ID: G-2604-2013

Храпова Наталия Игоревна, аспирант,
Юго-Западный государственный университет,
г. Курск, Российская Федерация,
e-mail: KhrapovaNI@yandex.ru,
ORCID: http://orcid.org/0000-0001-7947-1427

Супрунова Ольга Геннадьевна, аспирант,
Юго-Западный государственный университет,
г. Курск, Российская Федерация,
e-mail: Dlya.dela2010@yandex.ru,
ORCID: http://orcid.org/0000-0002-8371-6543

Дородных Александр Алексеевич, аспирант,
Юго-Западный государственный университет,
г. Курск, Российская Федерация,
e-mail: alex.dorodnych@mail.ru,
ORCID: http://orcid.org/0000-0003-0292-3127