М.Д. Долгушин, М.С. Цыганова

Тюменский государственный университет

УДК 519.688

РАЗРАБОТКА И РЕАЛИЗАЦИЯ АЛГОРИТМА ЦИФРОВИЗАЦИИ СХЕМЫ ТРУБОПРОВОДА

Аннотация. В статье представлен алгоритм построения цифровой модели трубопровода по его изображению на топографической карте, а также программный продукт, реализующий этот алгоритм. Полученная цифровая модель будет использоваться для определения числовых характеристик, необходимых в проектной документации.

Ключевые слова: цифровая модель, распознавание изображений, схема трубопровода, цветовая модель, цветовая фильтрация.

Постановка задачи

Решение различных задач, связанных с построением цифровой экономики, сопровождается увеличением количества электронной документации, подлежащей обработке. Это, в свою очередь, делает актуальной проблему внедрения элементов автоматизации в обработку поступающих документов. Примером может стать автоматизация обработки изображений схем трубопроводов, являющихся частью проектной документации в нефтегазовой отрасли.

Проектная документация по прокладке, реконструкции, текущему ремонту и т. п. трубопроводов включает как текстовую, так и графическую часть. Одной из задач подготовки комплекта документов по проекту является проверка соответствия схем трубопроводов, представленных на топографических картах, числовым характеристикам, содержащимся в текстовой части документации.

Основная цель данной работы – построение цифровой модели схемы трубопровода на основе имеющегося изображения на топографической карте. Полученная модель будет использоваться для определения числовых характеристик (длина трубопровода и отдельных его участков) и сравнения этих значений с параметрами, указанными в текстовой части документации. Реализация модели и ее численного анализа позволит автоматизировать поиск несоответствий в текстовых и графических документах.

Определение характеристик, идентифицирующих трубопровод

На первом этапе разработки модели были проанализированы схемы трубопроводов, содержащиеся в графической части документации ряда проектов. Для анализа использовались документы из открытого источника [1]. В процессе анализа было установлено, что не существует единообразной системы условных обозначений, используемой во всех проектах. Отсюда был сделан вывод, что алгоритм оцифровки схемы трубопровода должен выстраиваться с учетом особенностей карт различных типов. В рамках настоящей работы рассматривались два типа топографических карт, примеры которых взяты из [2] и [3]. Для каждого из этих типов был определен собственный набор значений характеристик, отличающий изображение трубопровода от других линий на карте (дорог, рек, выносных линий и др.).

Характеристики трубопровода топографических карт первого типа:

1. Насыщенность цвета: от 76% до 100%.
2. Яркость цвета: от 76% до 96%.
3. Тон цвета: любой.
4. Форма: одна прямая или кривая линия.
5. Толщина: 5 и более пикселей, слабо различается на разных участках трубопровода.

Характеристики трубопровода топографических карт второго типа:

1. Насыщенность цвета: от 0% до 16%.
2. Яркость цвета: от 0% до 16%.
3. Тон цвета: любой.
4. Форма: множество пересекающихся прямых, кривых; имеется условное обозначение в виде одноцветного круга.
5. Толщина: различается на разных участках трубопровода.

Выбор цветовой модели

Поскольку фильтрация изображений топографических карт по тону цвета является ресурсоемкой процедурой, использование цветовой модели RGB, основанной на соотношении тонов цвета в пикселе, на этапе фильтрации является нецелесообразным. Поэтому было принято решение использовать другую цветовую модель – HSV.

HSV – цветовая модель, в которой координатами цвета являются цветовой тон, насыщенность и яркость, поэтому фильтрация в координатах данной цветовой модели будет значительно эффективнее, с учетом того, что насыщенность и яркость являются известными параметрами в представленных наборах характеристик.

Преобразование из RGB в HSV проводится по следующим принципам:

$$H \in [0,360);\ S,V,R,G,B \in [0,1]$$

$$Max = \max\{R,G,B\};\ Min = \min\{R,G,B\}$$

$$H = \begin{cases} 0, \text{если } Max = Min \\ 60 * \frac{G-B}{Max-Min} + 0, \text{если } Max = R \text{ и } G \geq B \\ 60 * \frac{G-B}{Max-Min} + 360, \text{если } Max = R \text{ и } G < B \\ 60 * \frac{B-R}{Max-Min} + 120, \text{если } Max = G \\ 60 * \frac{R-G}{Max-Min} + 240, \text{если } Max = B \end{cases} \quad (1)$$

$$S = \begin{cases} 0, \text{если } Max = 0 \\ 1 - \dfrac{Max}{Min}, \text{в пр. случае} \end{cases} \qquad V = Max$$

Основная схема алгоритма и используемый инструментарий

Основные шаги разработанного алгоритма оцифровки схемы трубопровода показаны на рис. 1. Далее эти шаги будут рассматриваться совместно с их реализацией.

Реализация выполнена в виде программного консольного приложения. Предполагается, что пользователи приложения являются специалистами в соответствующей предметной области, и работа с приложением не составит для них труда. В качестве инструментов реализации были использованы язык программирования Python, а также библиотеки NumPy [4], Open CV [5] и Pillow [6].

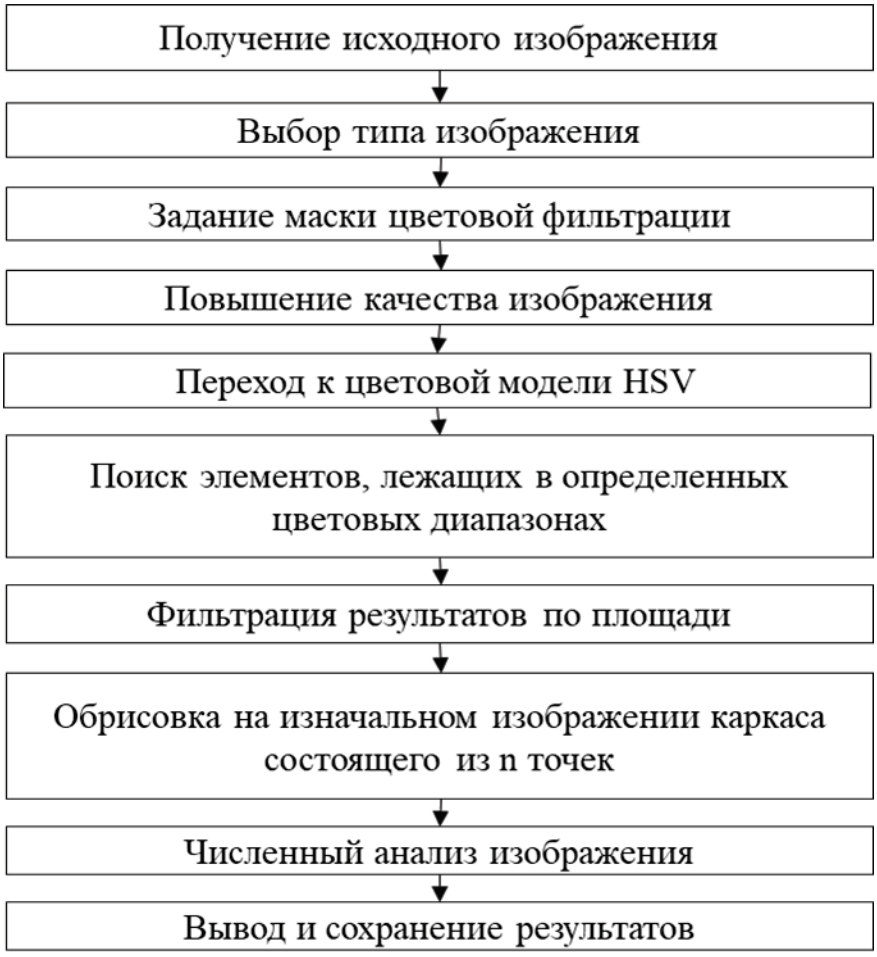

Рис. 1. Схема алгоритма оцифровки трубопровода.

Получение исходного изображения

В качестве исходного изображения используется цветное растровое изображение различных разрешений в различных форматах (PNG, BMP, JPG). Цветовая схема может быть представлена Grayscale (8 бит), RBG (24 бита) или sRGB (32 бита). Таким образом, в качестве входных данных можно использовать самые распространенные форматы изображений. В реализации используется функция из библиотеки Open CV:

cv2.imread(filename[, flags]) → retval

которая возвращает NumPy массив, содержащий цифровое представление данных изображения.

Выбор типа изображения

На этом этапе производится пользовательский выбор типа топографической карты посредством интерфейса приложения. В данной работе реализован выбор одного из двух вариантов, описанных на этапе выделения характеристик изображения трубопровода. Пользовательский выбор определяет параметры сегментации изображения, которые будут применяться на следующих этапах обработки.

Задание маски цветовой фильтрации

На данном шаге определяются верхние и нижние пороговые значения маски цветовой фильтрации в соответствии с характеристиками типа, выбранного ранее. Для первого типа нижние пороговые значения равны [0, 195, 195], а верхние – [255, 255, 245]; для второго типа, нижние пороговые значения равны [0, 0, 0], верхние – [255, 40, 40]. В массивах пороговых значений первая характеристика – тон цвета, вторая – насыщенность, третья – яркость, соответствующие значениям координат цветовой модели HSV.

Повышение качества изображения

На данном этапе из исходного изображения (карты) удаляются шумы, и повышается четкость выделяемого изображения трубопровода с помощью медианного фильтра. Идея алгоритма медианного фильтра [7] следующая. Для каждого пикселя изображения выполняется сортировка соседних ему пикселей (матрицы 3×3, в центре которой – текущий пиксель), после чего выбирается среднее значение из отсортированного массива. Это значение и является выходным для текущего пикселя.

Реализация выполнена с использованием функции библиотеки Open CV:

cv2.medianBlur(src, ksize[, dst]) → dst

которая возвращает NumPy массив, содержащий представление данных изображения, обработанное медианным фильтром**.**

Переход к цветовой модели HSV

Перевод изображения в цветовую модель HSV приводит к усилению тональной составляющей в цветных изображениях, что способствует выделению изображения трубопровода на карте. Основные принципы преобразования из цветовой модели RGB в HSV были представлены в (1).

В реализации используется функция библиотеки из Open CV:

cv2.cvtColor(src, code[, dst[, dstCn]]) → dst

которая выполняет преобразование из одной цветовой модели в другую.

Поиск элементов, лежащих в определенных цветовых диапазонах

На этом этапе выполняется поиск элементов изображения, лежащих в цветовых диапазонах искомых трубопроводов. Эти цветовые диапазоны были получены на этапе задания маски цветовой фильтрации.

В программной реализации используется функция библиотеки Open CV:

cv2.inRange(src, lowerb, upperb[, dst]) → dst

которая принимает NumPy-массив, содержащий цифровое представление

изображения, NumPy-массив нижнего цветового порога, и NumPy-массив верхнего цветового порога, а возвращает NumPy-массив, имеющий те же размеры, что и входное изображение, но с элементами цвета, удовлетворяющего заданным пороговым значениям цвета.

Фильтрация результатов по площади

В результате цветовой фильтрации могли остаться фрагменты изображения, не относящиеся к трубопроводу; для их удаления применяется дополнительная фильтрация по площади. Это возможно благодаря тому, что площадь части элементов трубопровода известна по выделенным ранее характеристикам.

На этом этапе работы алгоритма пользователю предлагается задать длину стороны фильтрующего по площади квадрата и минимальное заполнение (%) фильтрующего квадрата элементами из массива, прошедшего цветовую фильтрацию. Координаты центральных элементов успешно отфильтрованных квадратов добавляются в новый массив, который и является основным результатом оцифровки изображения.

В программной реализации разработана функция filterOnTough(rad, imsize, mask, ratio). Блок-схема алгоритма работы этой функции представлена на рис. 2. На вход подаются значения: rad – половина стороны квадрата фильтрации, imsize – массив, содержащий длину и ширину исходного изображения, mask – многомерный массив, содержащий представления изображения, обработанного цветовым фильтром, ratio – коэффициент заполнения квадрата фильтра удовлетворяющими условию цветового фильтра пикселями.

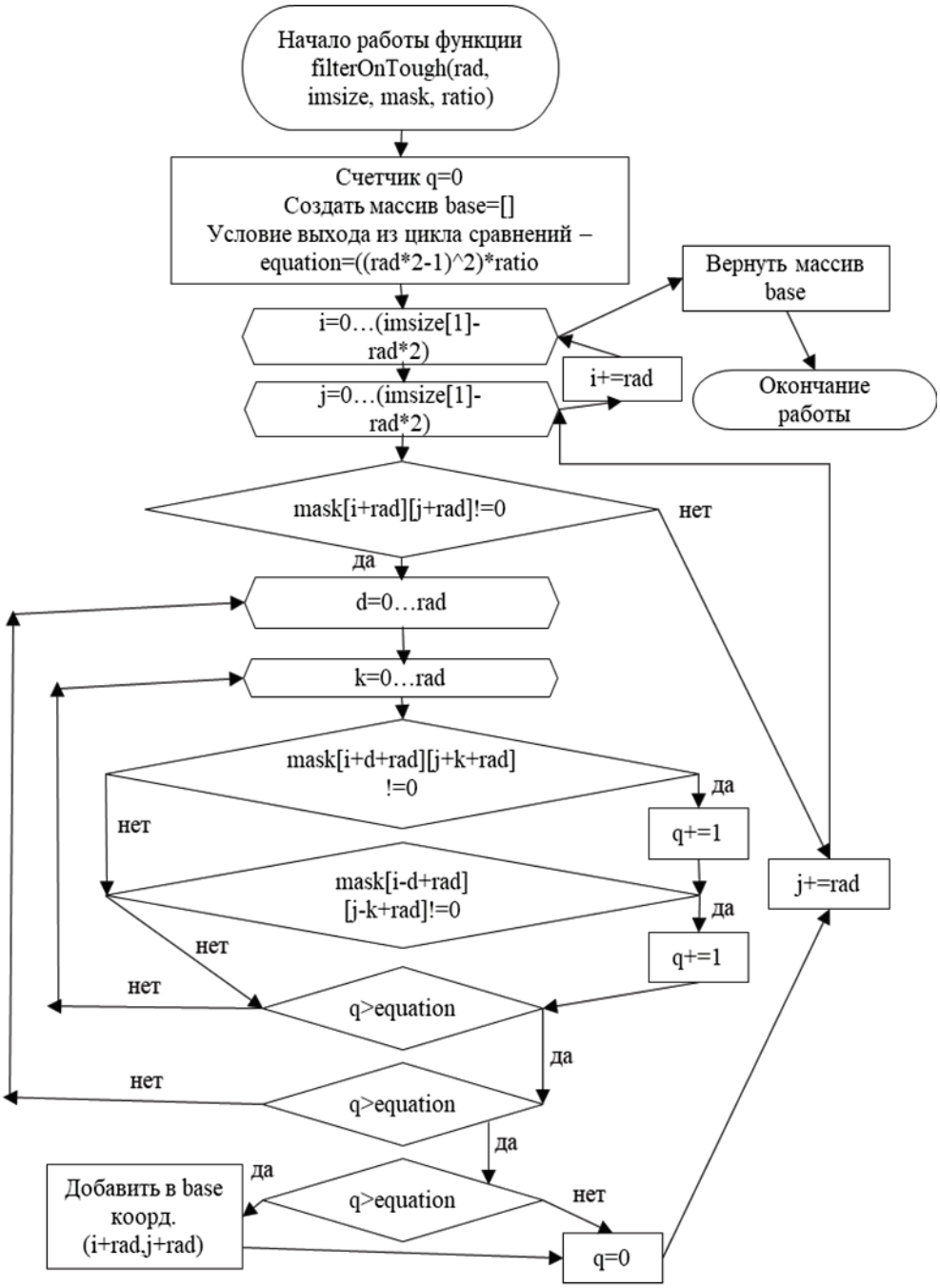

Рис. 2. Блок-схема функции filterOnTough().

Обрисовка каркаса из n точек

На этом этапе работы алгоритма пользователю предлагается задать в интерфейсе приложения количество точек, случайно выбираемых из полученного на предыдущем этапе массива, которые будут отмечены «поверх» исходного изображения и соединены линиями. Эти случайно выбранные точки будут также использованы на этапе численного анализа

изображения. Ограничение на число точек принимается для увеличения быстродействия алгоритма.

Отметка точек на изображении и соединение их линиями производится с помощью функции библиотеки Open CV:

cv2.line(img, pt1, pt2, color[, thickness[, lineType[, shift]]]) →None.

Здесь img – NumPy-массив, представляющий изображение, pt1 – координаты точки начала линии, pt2 – координаты точки конца линии, color – цвет линии в BGR, thickness – толщина линии, lineType – тип линии, shift – количество дробных бит в координате точки.

Численный анализ изображения

В данной работе численный анализ результатов, полученных после всех предыдущих этапов обработки, представлен в двух вариантах, соответствующих двум различным типам изображений, анализируемых разработанным приложением. Это связано с тем, что описанный выше алгоритм анализирует изображение трубопровода, представленного одной линией, (не рассматривает схему как множество пересекающихся или не пересекающихся линий).

Для изображений первого типа n случайно выбранных наборов координат точек сортируется по оси абсцисс, а затем для каждой точки, кроме первой, находится векторное расстояние до предыдущей точки, после чего все полученные значения расстояний складываются, и результат выводится на экран в виде консольного текста.

Для изображений второго типа находится число черных кругов, представляющих собой некоторые связующие узлы трубопровода. Точки изображения, прошедшие фильтрацию и находящиеся на расстоянии 15 px^2 друг от друга, принимаются за части одного узла. Следует отметить, что количество узлов трубопровода может быть найдено некорректно, если параметры фильтрации по площади были выбраны неверно, либо в том случае, когда в число точек, выбранных для каркаса, не попали точки

некоторого из кругов. В дальнейшем планируется доработать алгоритм, освободив его от указанного недостатка.

Вывод и сохранение результатов

Полученное на этапе отрисовки каркаса изображение выводится на экран в виде окна. В текущей версии приложения на экран также последовательно выводятся исходное изображение, изображение после медианной фильтрации, изображение, представленное в HSV, изображение после фильтрации по цвету, изображение после фильтрации по площади и только затем изображение с выделенным каркасом.

Вывод изображения на экран производится с помощью двух функций библиотеки Open CV:

cv2.imshow(winname, mat) → None

которая выводит NumPy-массив mat, представляющий изображение, на экран в виде диалогового окна;

cv2.waitKey([delay]) → retval

которая ожидает нажатия любой кнопки, предотвращая закрытие окон с выведенными изображениями.

Далее в интерфейсе предоставляется возможность сохранения изображения с выделенным каркасом в папку внутри приложения. Это реализовано с помощью двух функций модуля Image библиотеки Pillow:

PIL.Image.fromarray(obj, mode=None) –

преобразует поступивший NumPy-массив в изображение,

PIL.Image.save(fp, format=None, **params) –

сохраняет изображение по пути fp (объект типа pathlib.Path) в формате, заданном в пути, если это возможно.

Примеры выводимых окон представлены на рис. 3 и 4.

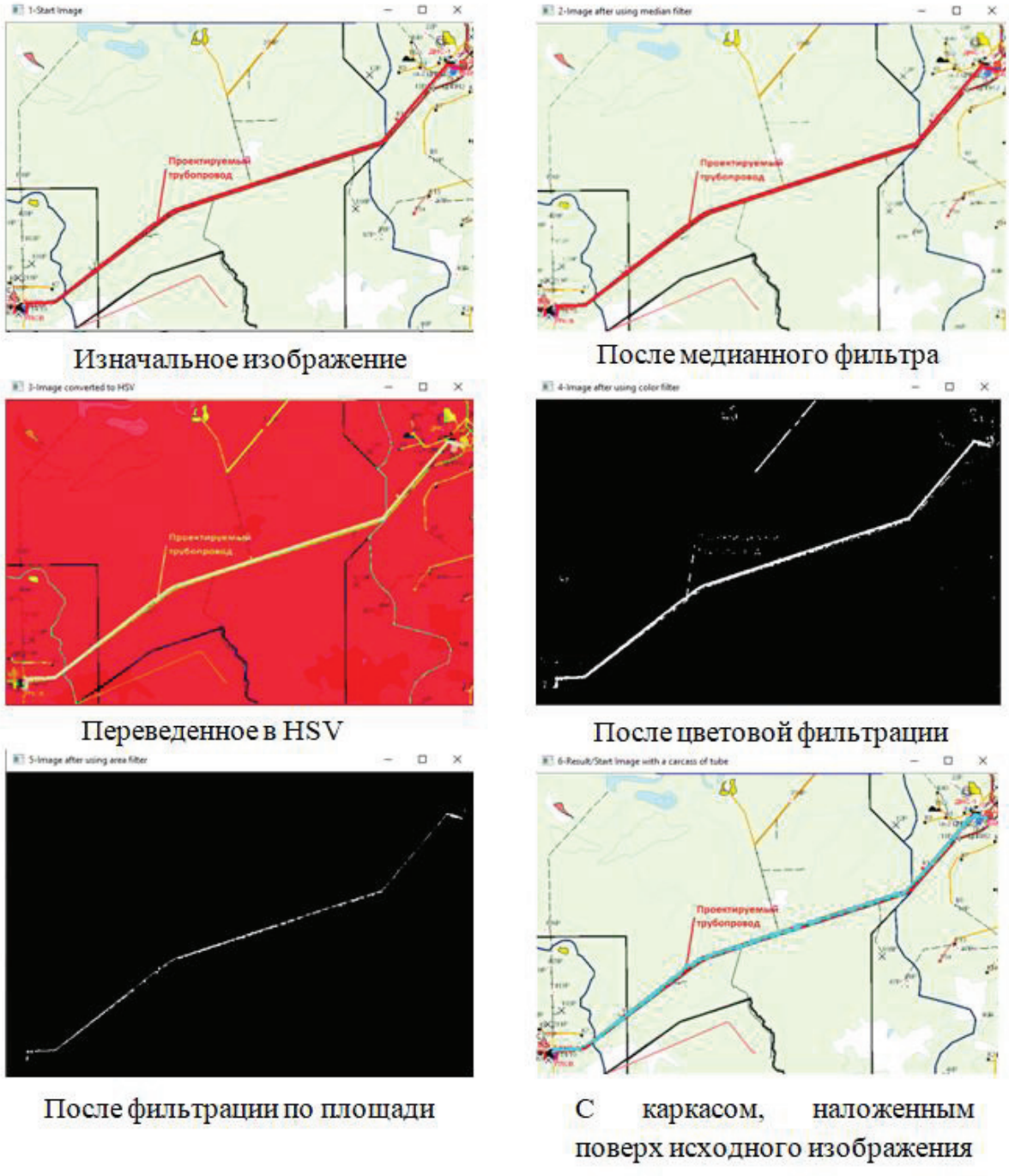

Рис. 3. Изображения, выводящиеся для карт первого типа.

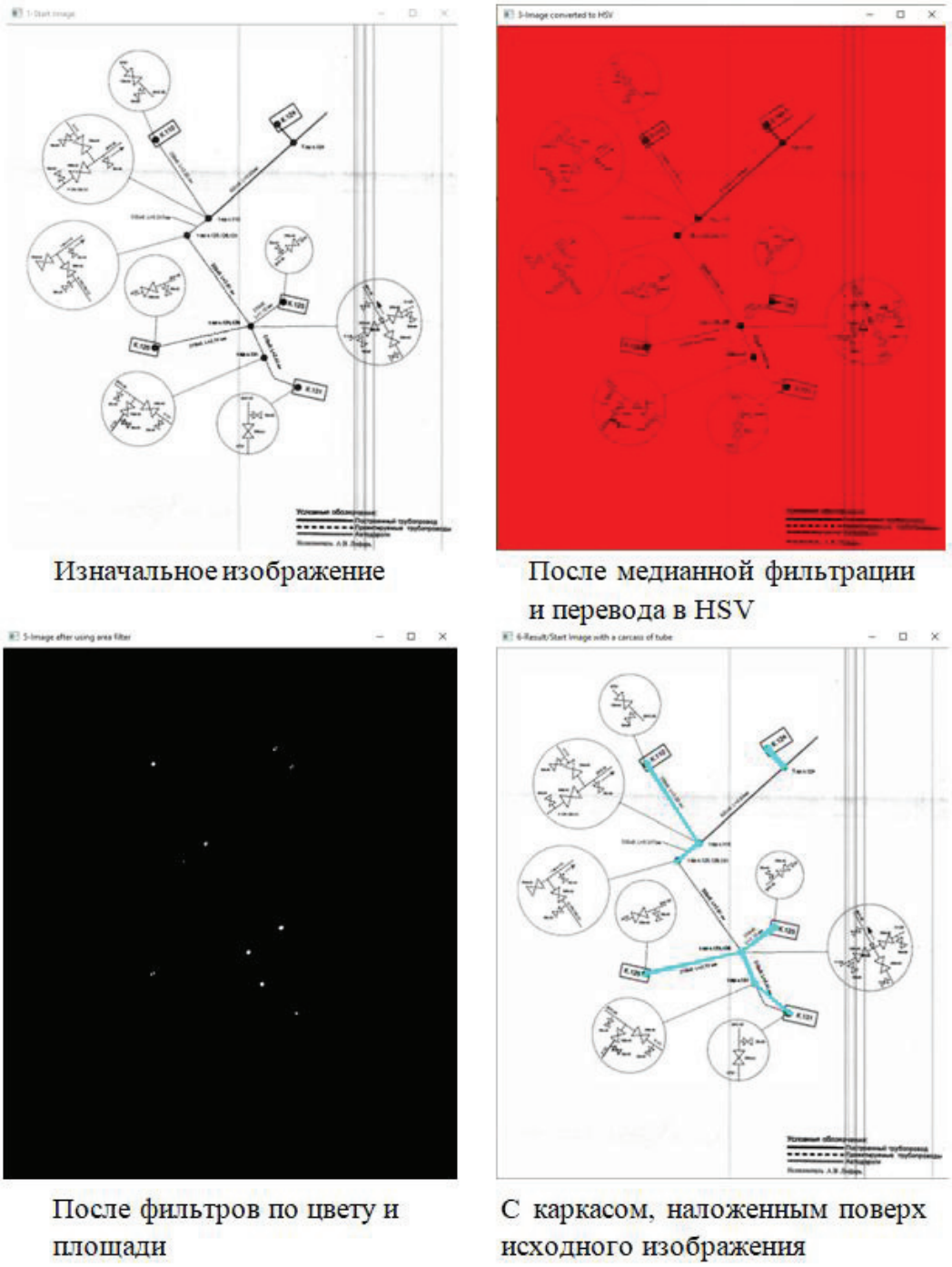

Рис. 4. Изображения, выводящиеся для карт второго типа.

Пример результатов взаимодействия с консолью приложения представлен на рис. 5.

```
Труба перового типа[1] или второго[2]?
1
Введите n
100
Введите длину стороны квадрата фильтрации(px)
5
Введите плотность заполнения квадрата фильтрации(%)
88
Длина трубопровода с избражения первого типа = 685.3477265215001 px
```

Рис. 5. Пример взаимодействия с консолью.

Заключение

Описанный в работе алгоритм позволяет выделять на топографических картах изображение трубопровода, представленного одной линией. Полученные результаты служат основой для определения числовых характеристик трубопровода и последующей проверки корректности проектной документации.

Дальнейшее развитие алгоритма оцифровки предполагает возможность анализа изображений трубопроводов различных форм (представленных множеством линий) в удобном для пользователя режиме.

СПИСОК ЛИТЕРАТУРЫ

1. Федеральная электронная площадка ТЭК-Торг // URL: https://www.tektorg.ru/
2. Извещение о закупочной процедуре PH816228 // URL: https://www.tektorg.ru/rosneft/procedures/70152
3. Извещение о закупочной процедуре 31806929128 // URL: https://www.tektorg.ru/rosneft/procedures/106982
4. Документация библиотеки NumPy // URL: https://numpy.org/doc/
5. Документация библиотеки Open CV // URL: https://docs.opencv.org/

6. Документация библиотеки Pillow // URL: https://pillow.readthedocs.io/en/stable/
7. UnickSoft. Матричные фильтры обработки изображений. URL: https://habr.com/ru/post/142818/