ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ»
(Н И У « Б е л Г У »)

ИНСТИТУТ ИНЖЕНЕРНЫХ ТЕХНОЛОГИЙ И ЕСТЕСТВЕННЫХ НАУК

КАФЕДРА МАТЕМАТИЧЕСКОГО И ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ ИНФОРМАЦИОННЫХ СИСТЕМ

РАЗРАБОТКА АВТОМАТИЗИРОВАННОЙ СИСТЕМЫ ГЕМАТОЛОГИЧЕСКОГО АНАЛИЗА НА ОСНОВЕ КРИВОЙ ПРАЙС-ДЖОНСА

Выпускная квалификационная работа
обучающейся по направлению подготовки 02.04.01 Математика и компьютерные науки
очной формы обучения, группы 07001631
Сойниковой Екатерины Сергеевны

Научный руководитель
к.т.н., доцент
Михелев В.М.

Рецензент
Профессор кафедры ПИиИТ,
д.т.н., доцент Черноморец А.А.

БЕЛГОРОД 2018

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

В настоящее время интенсивно развиваются интеллектуальные системы, предназначенные для автоматической обработки медицинских изображений. Автоматизированная обработка и анализ медицинских изображений являются универсальным инструментом медицинской диагностики [1][2]. Современные системы гематологического анализа позволяют выполнить предварительную классификацию клеток крови. Например, такие системы эффективно сортируют лейкоциты (8–10% ошибок) и обеспечивают более детальную информативность анализов крови [3][4].

Классификация клеток крови на микроскопическом изображении представляет собой, в терминах компьютерного зрения, задачу распознавания объектов.

Кровь – сложная функциональная система, обеспечивающая своевременную доставку кислорода и питательных веществ клеткам тканей и удаление продуктов метаболизма из органов и интерстициальных пространств [5]. Система крови тонко реагирует на воздействия факторов среды набором специфических и неспецифических компонентов [11]. Важная характеристика физиологии и патологии системы крови – количественный и качественный состав эритроцитарной популяции [6].

Визуальная оценка морфологических характеристик клеток крови является неотъемлемой частью анализа крови человека. Определение количества форменных элементов крови разного типа, их соотношения является важным и наиболее частым тестом клинической лабораторной диагностики.

Исторически идентификация и счет клеток крови производились с использованием микроскопа в «ручном» режиме, при этом исследуемый образец крови находился в статическом состоянии.

В последние годы интенсивно развивается иной подход к идентификации и счету форменных элементов крови – метод цифровой микроскопии. В настоящее время это перспективное направление находится в стадии проработки, поиска соответствующих оптимальных алгоритмов и программ для минимизации ошибок при счете форменных элементов крови. В работах [2]-[4] рассмотрена задача подсчета эритроцитов на изображениях препаратов крови, полученных с помощью цифрового микроскопа. Достигнутая точность счета эритроцитов по отношению к «ручному» способу составляет 96–98 %.

Автоматическая сегментация и счет эритроцитов на основе анализа микроизображений клеток крови выполнены в работах [16]-[18]. Более сложной по сравнению с идентификацией и счетом эритроцитов является задача распознавания лейкоцитов. В работе [10] представлен алгоритм, позволяющий вести счет клеток крови с учетом сложности и произвольности их формы.

Таким образом, цель работы – это разработка принципов идентификации и счета эритроцитов, на основе цифровой микроскопии. Решение задач, позволяющих обеспечить достижение этой цели является актуальным и востребованным в настоящее время.

В данной работе рассматривается задача постановки возможного диагноза по гематологическому анализу цифрового изображения эритроцитов. Описываются шаги по предварительной обработке изображения для уменьшения шумов и точности сегментации объектов клеток на классы. Для каждого этапа приведены примеры работы фильтров. Заключительным шагом является построение гистограммы распределения площадей объектов клеток – кривая Прайс-Джонса. По форме гистограммы можно предположить о наличии заболевания у человека, у которого взяли препарат крови на исследование. В заключении статьи приводятся описания типичных заболеваний, которые могу быть выявлены с помощью такого рода

анализа. Для реализации анализа реализована программа на языке Python 2.7 с библиотеками OpenCV и Seaborn.

В первой главе производится обзор гематологического анализа медицинских изображений. Представлено несколько видов сегментации изображений, отмечены свои плюсы и минусы. Доступным текстом представлена информация о гематологическом анализе медицинских изображений. Представлены эталонные кривые Прайс-Джонса здорового и больного человека. Произведен обзор областей применения компьютерной обработки изображений. Также проанализированы проблемы, с которыми можно столкнуться во время обработки изображений. Для подтверждения актуальности рассмотрено современное состояние и критичные недостатки текущего состояния системы. Конкретно выделена цель и задачи, которые полностью соответствует теме выпускной квалификационной работе.

Во второй главе рассматриваются основные математические операции, которые являются главным инструментом в обработке изображений. В каждой функции OpenCV реализован сложный математический аппарат, в данной главе раскрывается математику каждой использованной функции, которая входит в разработанный алгоритм.

В третьей главе идет описание разработанного программного обеспечения. Наиболее важные части кода (функции обработки изображений) представлены на листингах, также после описания разработки каждого этапа алгоритма приведены примеры на рисунках.

В четвертой главе произведен вычислительный эксперимент работы алгоритма.

В заключении сделан вывод о степени достижения поставленных целей и задач.

Данная выпускная квалификационная работа содержит 62 страницы, 30 рисунков, 1 таблицу, 9 формул, 8 листингов кода и 1 приложение.

1 ОБЗОР ТЕМЫ ГЕМАТОЛОГИЧЕСКОГО АНАЛИЗА МЕДИЦИНСКИХ ИЗОБРАЖЕНИЙ

1.1 Гематологический анализ

Гематологический анализ – это исследование, которое чаще всего назначает врач при первичном обследовании пациента. Самый простой и легко осуществимый способ узнать о неполадках в организме и понять, в какую сторону двигаться дальше – сдать кровь на гематологию. Сделать это можно во всех без исключения муниципальных поликлиниках, больницах и платных медицинских центрах.

Гематологический анализ – это описание самых важных компонентовкрови, дающее представление о наличии воспалительных и онкологических процессов. В ходе анализа изучаются все клетки, входящие в состав крови, определяется их размер, масса, количество и процентное соотношение. Кроме того, измеряется уровень содержания гемоглобина, показатель гематокрита и скорость реакции оседания эритроцитов.

Для проведения представленного анализа в качестве материала используется так называемая контрольная кровь, представляющая собой смешение кровяных клеток, консервантов и специальной жидкости, напоминающей по своему составу плазму [2]. Забор крови для основы может быть взят как из вены пациента, так и из безымянного пальца. Во многом это зависит от основной цели исследования.

Процедура проходит в лабораторных условиях, она практически безболезненна и занимает не более получаса. После этого биологический материал передается в лабораторию, где исследуется при помощи специального лабораторного прибора – гематологического анализатора.

Происходит это следующим образом. В электрический аппарат, заправленный особого вида реактивами, помещается пробирка с контрольной кровью и буквально за считанные минуты устройство выдает бланк с распечатанными результатами, к которым относятся: состояние кровяных телец, ретикулоцитов, гемокрит, уровень содержания гемоглобина и общая лейкоцитарная формула, содержащая точные сведения вплоть до количества гранулоцитов [12].

Наиболее мощные и дорогостоящие гематологические анализаторы в числе прочего определяют степень содержания в крови пациента лимфоцитов, моноцитов, нейтрофилов, эозинофилов, а также их состояние. Представленное устройство отличается повышенной точностью и надежностью [10].Кроме того использование оборудования значительно облегчает работу врача и позволяет сэкономить немало времени, поскольку в этом случае результаты анализа выдаются буквально через пять-десять минут.

1.2 Компьютерное представление и обработка изображений

Цифровое изображениепредставляет собой массив данных, который получен с помощью перевода исходного непрерывного сигнала в цифровой. Эта процедура называется дискретизацией. Существует два вида представления цифровых изображений: растровое и векторное.

Растровое изображение это изображение, которое содержит в себе множество цветных точек. В компьютере растровое изображение представляется в виде матрицы, которая заполена элементами изображения. Элементы изображения называют пикселами. [8] Каждый отдельный пиксель изображения представляет собой число,в двоичной системе счисления, которое показывает измеренное значение яркости нужного цвета.

Растровое изображение содержит в себе свойства, которые являются основными характеристиками изображения, такие как размер и глубина цвета.

Размер изображения — это размерность матрицы с пикселами изображения[22].

Глубина цвета показывает сколько бит выделено на пиксель. По глубине цвета можно сказать, сколько цветов содержится в изображении.

Таблица 1.1

Соответствие значения глубины цвета и количество цветов в палитре

Глубина цвета, I (битов)	Количество цветов в палитре, N
4	2^4=16
8	2^8=256
16	2^{16}=65 536
24	2^{24}=16 777 216

Таблица 1.1. показывает какое количество цветов можно выделить, при разных значениях глубины.

Основной недостаток пиксельного изображения состоит в том, что размер пикселов является фиксированным. Из-за этого в случае изменения размера изображения возникают крайне нежелательные эффекты.

Векторное изображение состоит из множества геометрических объектов, которые формируются по математическим уравнениям. Геометрические объекты представляют собой точку, линию либо многоугольник.

Основой векторного изображения является векторный каркас.

Для того чтобы уменьшить или увеличить изображение необходимо всего лишь изменить параметр изображения - масштаб.[34] При значительном уменьшении или увеличении изображение данного типа не искажаются. Это можно считать главным достоинством векторного изображения. Еще одним достоинством изображения представленного в векторном виде является небольшой размер файла (100-200 кб).

На рис. 1.1 показана существенная разница между растровым и векторным представлением изображения.

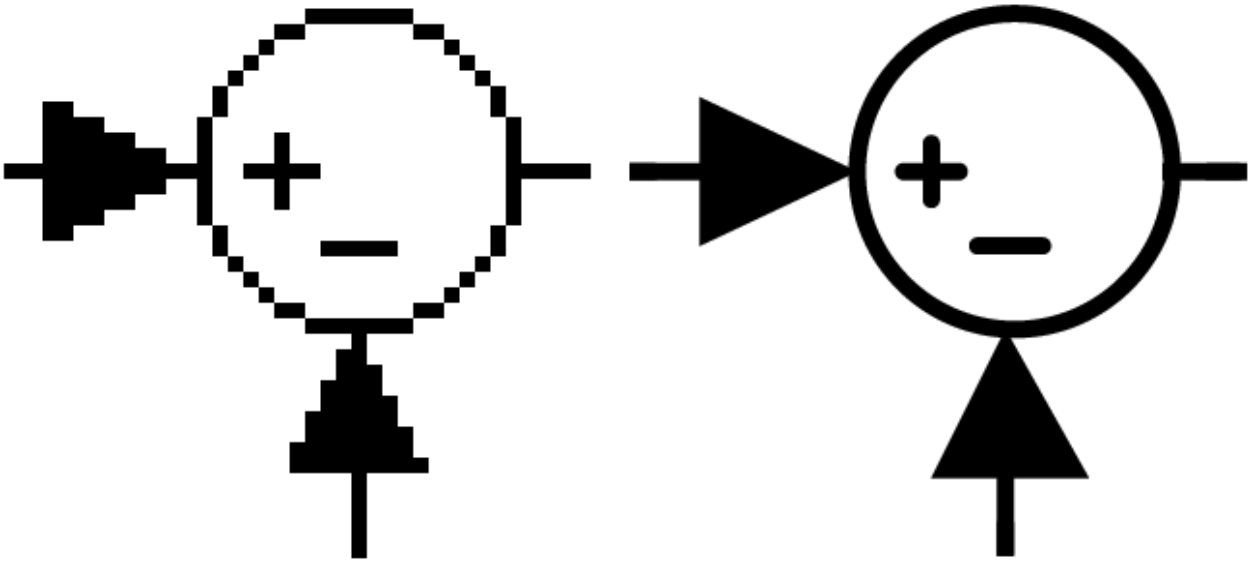

Рис. 1.1. Отличие растрового и векторного изображения

Поскольку растровое изображение представлено в виде таблицы пикселей, а не в виде математических формул, его используют в области цифровой обработки изображений.

1.3 Компьютерная обработка изображений. Области применениябработки изображений

Цифровой обработкой изображения называют цифровую обработку цифровых изображений с помощью вычислительных машин.

Цифровую обработку выполняют обычно над изображением представленном в растровом виде (изображение представлено как матрица пикселов). В процессе обработки изображений происходит преобразование матрицы пикселов, в итоге формируется новое обработанное изображение или частично обработанное изображение[40]. Преобразование указанной матрицы может производится над конкретными значениями элементов, над индексами элементов, над матрицей в целом либо над группой элементов.

После произведения изменения над элементами изображения получаем изображение размер которого совпадает с размером исходного изображения.

В ходе практической реализации определенных преобразований над элементами матрицы, новое значение пиксела вычисляется по конкретной функции.

В настоящее время расширяется область применения цифровой обработки изображения, тем самым заменяя аналоговую обработку сигналов изображений.

Цифровая обработка изображений охватывает широкие и разнообразные области применений. Она нашла широкое применение в медицине, искусстве, космосе, промышленности, а также в обнаружении и распознавания объектов.

В настоящее время в медицине применяются специализированные устройства преобразующие изображение в цифровую форму. Обработка медицинских изображений заключается в формировании изображений, улучшения качества и автоматической обработке. Медицинские снимки создаются с помощью электронных микроскопов, рентгеновских аппаратов, томографии, что и является предметом исследования в области цифровой обработки.

Для восстановления старых фильмов, производится процесс автоматического устранения дефектов в видео данных, которые получены путем преобразования киноизображения в видеоматериал.

Переданная цифровая информация с космических аппаратов требует выполнения передачи многочисленных потоков информации. Например, для передачи цифровой информации цветного телевидения, необходимо передавать потоки со скоростью 216 Мбит/с, а для передачи цифрового сигнала высокой четкости скорость передачи должна составлять 1Гбит/с.

Автоматический анализ нашел свое применение в области экспертизы, а также в области дистанционного наблюдения, при анализе местности, лестного хозяйства, для подсчета площади вырубок, для наблюдения созревания урожая [16].

На сегодняшний день невозможно представить области деятельности без компьютерной обработки. Техника, которая вошла в повседневное пользование, такая как интернет, телефон, принтер, сканер, фотоаппарат, видеокамера немыслима без компьютерной обработки изображений.

Компьютерная обработка изображения является одной из главных частей а областях деятельности, так как выполняет ряд важных задач: измерение параметров, улучшение качества изображений, распознавание изображений, сжатие изображений и др.

Также устройства преобразования изображении получили широкое распространение в областях техники, промышленности, медицине, науке и др. Назначение этих устройств решает ряд важных технических и научных задач, которые требуют анализа изображения.

Переход от аналоговой формы представления изображений к цифровой позволяет повысить и расширить применение алгоритмов для решения поставленной задачи.

1.4 Сегментация изображений

Сегментация - разбиение изображения на однородные области, схожие по какому либо критерию. В этих областях (сегментах) находятся пиксели похожие по некоторым характеристикам, например, по текстуре, яркости или цвету. Области, которые находятся по соседству будут значительно отличаться по заданной характеристике.

Главная задача сегментации заключается в том, чтобы преобразованное изображение было проще и легче анализировать, по определенным свойствам.

Сегментация используется в разных областях применения, но для каждой области необходим индивидуальный подход к обработке, всвязи с эти выделяют несколько видов сегментации.

Первый самый простой вид, это пороговая сегментация. Пороговая сегментация состоит в простом объединении близких по характеристикам областей изображения в небольшое число сегментов. Процедура заключается в том, что выбирается определенный порог (задается пользователем либо зависит о каких либо параметров), далее находится значение пикселя в матрице изображения. Если значение пикселя больше указанного порога, то вместо этого пикселя ставим -1, если меньше- 0 [17].

Следовательно, пороговую обработку можно получить из соотношения (1.1)

$$r(x,y) = \begin{cases} 1, & \text{при } s(x,y) > L \\ 0, & \text{при } s(x,y) < L' \end{cases} \qquad (1.1)$$

где$r(x,y)$ и $s(x,y)$ – уровни яркости пикселей изображения, L – пороговый уровень по яркости.

Данный вид сегментации содержит ряд недостатков:

- чувствителен к освещению;
- дает нечеткие контуры;
- чувствителен к шуму.

На рис. 1.2и рис. 1.3 представлены изображения до и после пороговой сегментации.

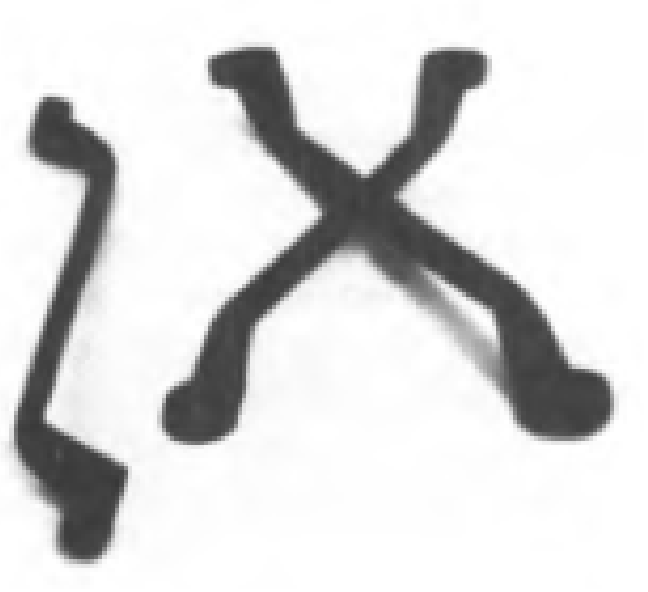

Рис.1.2. Исходное изображение

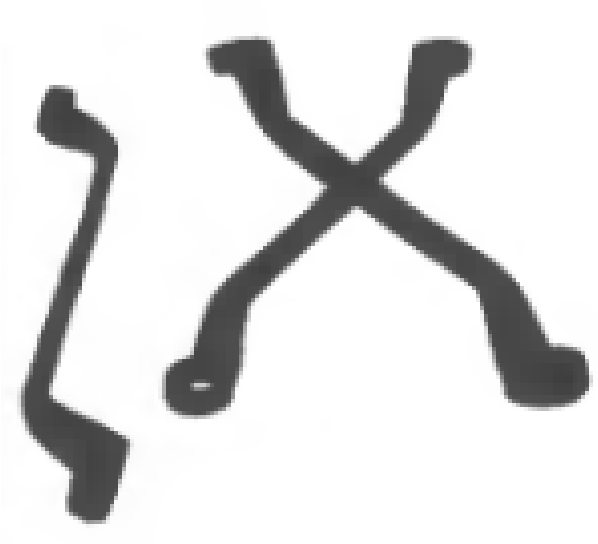

Рис.1.3. Изображение после пороговой сегментации

Второй вид сегментации, это выделение областей. Заключается в выделении однородных областей в изображении. Алгоритм поиска однородных областей предполагает сравнение начального пикселя и соседних с ним пикселем, для проверки близости их значений[31]. Если значения яркостей достаточно схожи, то эти пиксели объединяются в одну область. В результате область формируется за счет сращивания отдельных пикселей. Процесс сращивания продолжается до тех пор, пока области не пройдут все проверки на однородность.

На рис. 1.4 и рис. 1.5 представлены изображения до и после сегментации выделения областей.

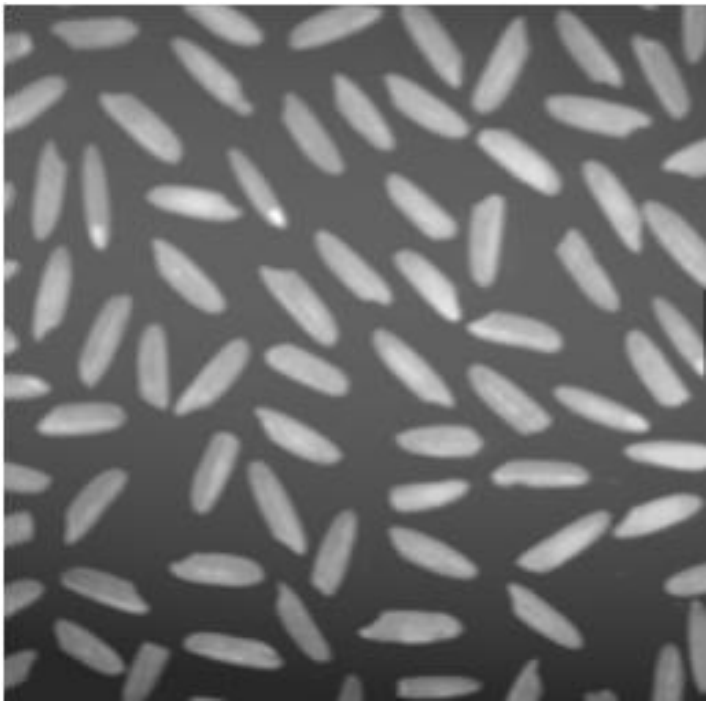

Рис.1.4. Исходное изображение

Рис. 1.5. Изображение после сегментации выделения границ областей

Следующий вид сегментации - алгоритмы выделения границ. Граница - это резкий переход яркости, следовательно, схема алгоритма заключается в обнаружении разрывов яркости. Для того чтобы найти резкое изменение яркости, необходимо найти абсолютное значение градиента, так как наибольшее изменение градиента происходит в направлении градиента [38].

Данные алгоритмы довольно точны, однако имеют большую вычислительную сложность.

На рис. 1.6 и рис.1.7 представлены изображения дои после сегментации выделения границ.

Рис.1.6. Исходное изображение

На рис. 1.7 представлено изображение после применения сегментации выделения границ.

Рис. 1.7. Изображение после пороговой сегментации

1.5 Проблемы обработки изображений

Классической задачей обработки изображений является улучшение качества. Эта задача впервые возникла в оптике и всегда решалась построением все более совершенных оптических систем, т.е. методами оптической обработки. С момента появления компьютеров в оптике произошла настоящая революция, связанная с проникновением в нее цифровых методов. Первые публикации по цифровой обработке изображений появились в конце 60-х годов, в основном применительно к задачам астрономии, радиофизики, биофизики, ядерной физики (пузырьковая камера) и целому ряду других прикладных задач.

Задача, о которой уже говорилось, - это задача повышения качества изображений. Она сейчас получает определенный новый импульс, и я также постараюсь показать некоторые результаты по улучшению качества изображений. Задача спектрального анализа многомерных сигналов - то, что говорилось по поводу быстрого преобразования Фурье и других алгоритмов, примыкающих к нему. Задача сжатия данных. Это очень актуальная задача в обработке изображений, связанная (особенно сейчас) с развитием телекоммуникационных систем [37].В них обязательно используется сжатие данных, поскольку если этого не делать, то мы столкнемся с тем, что время ответа системы будет очень большим. Поэтому огромные массивы данных, которыми характеризуется каждое изображение, обязательно подвергаются сжатию, или компрессии, как иногда говорят, и это используется, собственно говоря, повседневно всеми нами, когда мы работаем с персональным компьютером[2]. Задача формирования признаков в распознавании изображений. В распознавании образов есть две стороны, две проблемы. Одна - это отбор и упорядочение признаков; вторая - классификация образов. Первая сторона - отбор и упорядочение признаков - является неформальной, трудно формализуемой задачей. Надо глубоко понимать сущность

изображения, его природу, и тогда можно удачно выбирать признаки. Это позволит удачно решить задачу распознавания образов. И, наконец, задача синтеза оптических волновых полей, и в том числе компьютерная оптика. Это абсолютно новая вещь, связанная с тем, что можно с помощью компьютерного синтеза создавать новые оптические элементы, то есть фактически создавать новую элементную базу для оптических систем

При обработке изображений возникает ряд проблем, которые могут привести к отрицательным результатам в разных областях применения.

Одной из основных проблем обработки изображений, которая затрудняет анализ изображения, является шум. В изображении, которое получено посредством оцифровки, всегда присутствует посторонний шум, который ухудшает качество изображения. Шум образуется в изображении в виде случайно расположенных точек, которые по размеру похожи с пикселом. Цифровой шум отличается от изображения более светлым или темным оттенком серого.

При практической работе восстановления или реставрации изображения важное место занимает задача шумоподавления.

В компьютерных системах изображение представляется как цифровые потоки большой размерностью. Анализ изображений предполагает применение достаточно сложных функций для обработки каждого пикселя изображений[24].Отсюда следует, что сложность функций и большое разрешение изображений влечет за собой большие затраты машинного на вычисления. Это влечет за собой еще одну проблему в области обработки изображений, решением которой является формирование требуемых методов для проведения обработки, передаче и хранения необходимых объемов данных, которые связаны с изображениями разной природы. При программной реализации алгоритмов обработки изображений актуальным является использование технологий высокопроизводительных технологий, для решения вышеописанной проблемы.

1.6 Кривая Прайс-Джонса

Кривая Прайс-Джо́нса - кривая, отражающая распределение эритроцитов по их диаметру.

Одним из важных диагностических исследований для установления правильного и точного диагноза является измерение диаметра эритроцитов – эритроцитометрия.

Данные эритроцитометрии представляют в виде графика. Графическое изображение соотношения содержания в крови эритроцитов с различными диаметрами называют эритроцитометрической кривой Прайс-Джонса, где по оси абсцисс откладывают величину диаметра эритроцитов (в мкм), а по оси ординат — процентное содержание эритроцитов соответствующей величины. В процентном отношении диаметры эритроцитов у здоровых людей распределяются следующим образом: 5 мкм — 0,4% всех эритроцитов; 6 мкм — 4%; 7 мкм — 39%; 8 мкм — 54%; 9 мкм — 2,5%. У здоровых людей эритроцитометрическая кривая имеет правильную, с довольно узким основанием, почти симметричную форму. Виды кривой Прайс-Джонса можно увидеть на рис. 1.8 и на рис. 1.9.

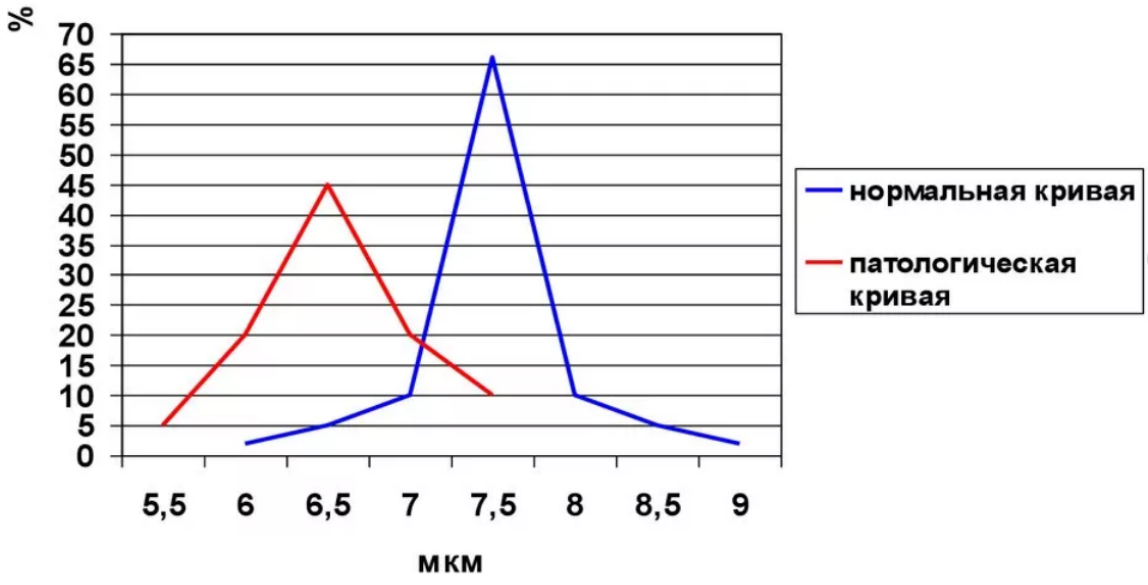

Рис. 1.8.Эритроцитометрическая кривая Прайс-Джонса

Результаты эритроцитометрии имеют важное значение для уточнения характера анемии. При наследственной анемии (микросфероцитозе), талассемии, железодефицитной анемии, свинцовом отравлении - как правило, выявляют большое количество мелких эритроцитов – микроцитоз, и соответственно, сдвиг кривой Прайс-Джонса влево. Увеличение количества крупных эритроцитов (макроцитов) — признак анемии при дефиците витамина В12 и фолиевой кислоты. При этих формах анемии кривая Прайс-Джонса имеет неправильную пологую форму с широким основанием и сдвинута вправо, то есть в сторону больших диаметров. Особенно важен такой анализ для уточнения диагноза анемии у детей[35].

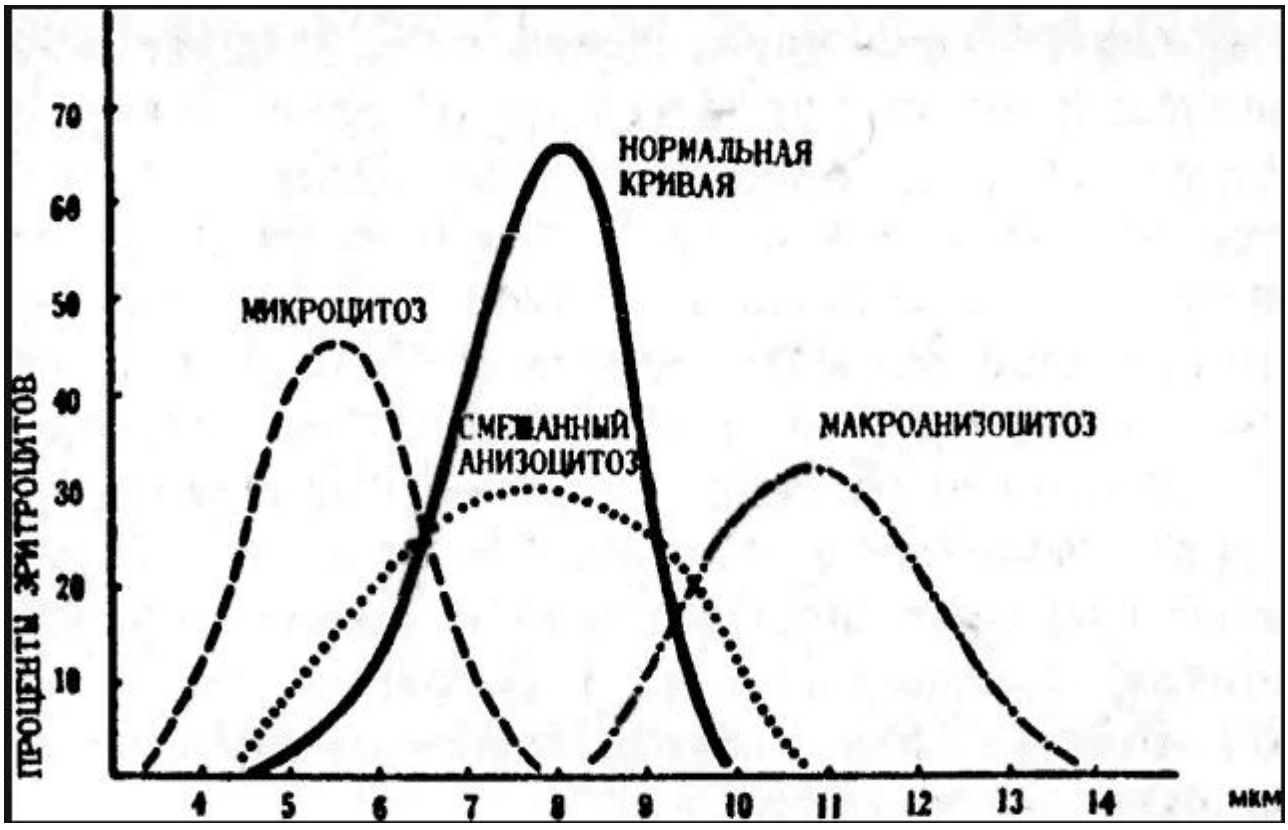

Рис.1.9. Болезни, которые показывает кривая Прайс-Джонса

1.7 Актуальность

Большое значение для успешного решения проблем, касающихся здоровья россиян, имеет современная и качественная лабораторная диагностика. В настоящее время значительная часть проектов направлена на повышение качества жизни больных, которые невозможны без использования высокотехнологичных приборов для автоматизированного анализа. В нашем случае, будет рассматриваться автоматизированный гематологический анализ клеток крови[6]. Технология автоматического гематологического анализа предполагает подсчет клеток и их геометрических характеристик.

При исследовании данной темы, было выявлено несколько проблем, касающихся в первую очередь обработке изображений, а также применению сложных математических функций при обработке изображений большого разрешения. Немаловажную роль занимает и быстродействие работы программы, так как предполагается работать с большими данными. Так как мы говорим о здоровье человека, а также о показателях здоровья человека, то самое главное значение в работе имеет точность обработки изображения и точность подсчета геометрических характеристик.

В связи с этим следует актуальность данной темы:

- Автоматизация – направление научно-технического прогресса, использующие технические средства и математические методы с целью освобождения человека от участия в обработке и анализе изображений.
- Точность в сегментации изображении, разбиение областей на классы.
- Точность в подсчете количества клеток, а также их площадей.

Объект – методы информационного анализа гематологического мазка крови.Предмет – методы построения кривой Прайс – Джонса.

1.8 Современное состояние

На кафедре биологии и экологии имеется оптический микроскоп, установлено ПО для обработки изображений, захваченных в обработки изображения, а также геометрические расчеты параметров о время сканирования мазка крови. ПО содержит процедуры морфологической клеток крови. Недостатком ПО является полуавтоматическая обработка изображений, т.е. лаборант вручную производит оконтуривание клеток крови. В связи с этим происходят неправильные расчеты площадей, следовательно, строится неправильно кривая Прайс-Джонса. Неточное построение кривой будет нести за собой неправильный диагноз пациента, а также неправильно подобранное лечение.

Современное состояние изображено на рис. 1.10.

Рис.1.10. Современное состояние

Исходя из этого, можно отметить, что современное состояние требует доработку и уточнение.

1.9 Постановка задачи

Цель работы: разработка автоматизированной системы гематологического анализа на основе кривой Прайс-Джонса.

Для решения поставленной цели необходимо решить следующие задачи:

- Провести исследования изображений клеток крови, полученных с помощью оптического микроскопа с целью выявления отличительных признаков, используемых врачом (биологом) при ручном анализе в процессе построения кривой Прайс-Джонса.
- Разработать алгоритм построения кривой Прайс-Джонса.
- Разработать программное обеспечение, реализующее описанные алгоритмы.
- Оценить работоспособность разработанного алгоритма на основе вычислительных экспериментов.

Классификация клеток крови на микроскопическом изображении представляет собой, в терминах компьютерного зрения, задачу распознавания объектов. Стандартный подход к решению этой задачи предусматривает следующие два этапа:

- разделение (сегментация) изображения на области, соответствующие объектам и фонhhy;
- непосредственно распознавание объектов, включающее в себя выделение характерных признаков объектов и распределение объектов в соответствии с их признаками по классам.

Специфика задачи сказывается на формировании векторов признаков объектов, на выборе метода сегментации, а также классификатора.

Таким образом, при выборе стратегии решения данной задачи особое внимание должно быть уделено изучению опыта применения различных методов сегментации микроскопических изображений и характерным признакам, используемым для классификации клеток крови.

Основным результатом сегментации микроскопического изображения мазка крови является выделение объектов интереса (клеток крови) с целью их дальнейшей классификации. Качество сегментации – ключевой фактор для получения адекватных значений характерных признаков объекта. Так, в работе [8] отмечено, что более половины ошибок классификации лейкоцитов были обусловлены неправильной сегментацией.

Основные причины, ведущиек ошибкам сегментации микроскопических изображений: перекрывание одной клетки другой, сильная вариация клеток по форме и размеру, воздействие разных факторов на внешний вид клетки, слабая контрастность изображений, зашумленность и артефакты на снимке препарата. Также влияет окраска препаратов крови: часто после окраски контрастность контуров структурных элементов внутри клетки превышает контрастность границ самой клетки, это может вызвать пересегментацию изображения, либо потерю части пикселей внутри объекта клетки.

Предлагаемый метод предварительной обработки изображений представлен на рис. 1.11.

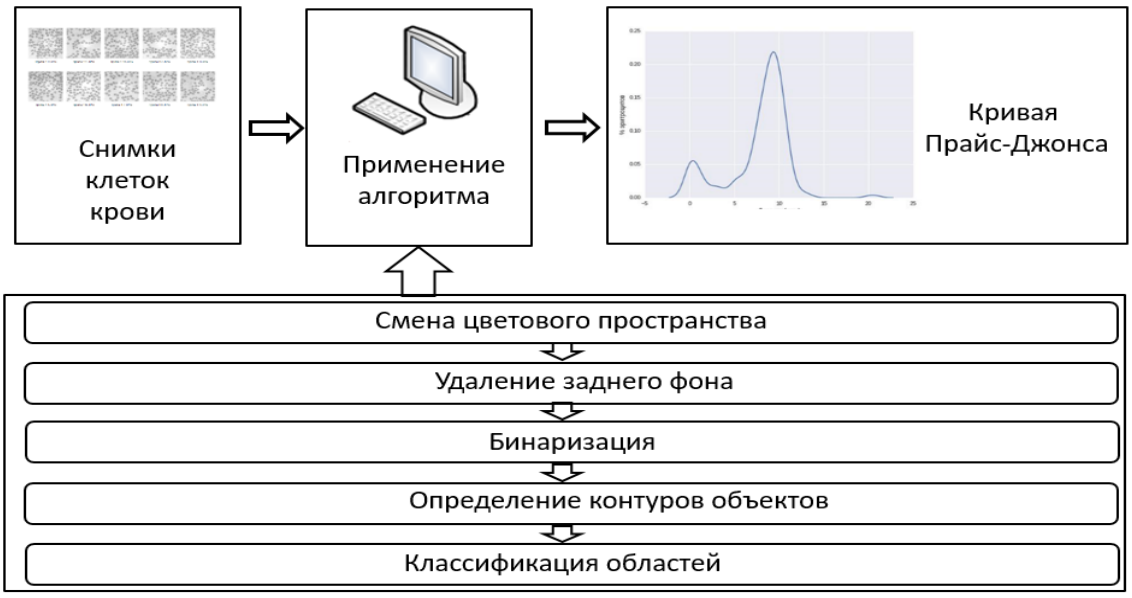

Рис.1.11. Предлагаемый метод

Основные этапы предложенного метода:

1. Смена цветового пространства**:** перевод изображения в цветовое пространство HSL (HLS в представлении OpenCV) с выделением отдельно канала L – светимость конкретного пикселя. Специфика исходных данных (изначально изображения представлены в оттенках серого) позволяет первым этапом выделить один канал и с ним дальше работать. Также, большинство методов OpenCV, необходимых для обработки изображения, тоже работают с одним каналом. В этом есть и доля оптимизации – нет необходимости на каждом этапе L-канал из изображения и в конце восстанавливать канальность изображения в исходное состояние, за исключением методов работы с многоканальными изображениями, но, в любом случае, это вычислительно дешевле, нежели выделение и пере сборка изображения на каждом этапе.
2. Удаление заднего фона: в некоторых местах есть области пересвечивания пикселей и наоборот – затемнения, из-за которых происходит неправильное контурирование клетки. Так как изначально

изображение зашумлено, необходимо также сгладить перепады между группой соседних пикселей.Сглаживание изображения просто сделать через размытие по Гауссу. Ядра размером 3x3 вполне достаточно, чтобы убрать неравномерности в шумах фона, но в то же время не потерять границы объектов.После выравнивания фоновых пикселей изображения необходимо увеличить контрастность изображения, чтобы при удалении фоновых пикселей оставить пиксели объектов, которые примерно равны по светимости фоновым.Отличие CLAHE от других алгоритмов в том, что он выполняет эквализацию на ограниченной области изображения с предварительно ограниченной гистограммой светимости. Таким образом, он более устойчив к порождению шума на гомогенных областях.

3. Бинаризация изображения: следующим шагом необходимо выполнить бинаризацию изображения, то есть приведение изображение к виду, когда каждый пиксель кодируется либо единицей, либо нулем. Этот шаг необходим, так как некоторые последующие функции OpenCV используют бинарное изображение, как один из аргументов.Выбор порога сегментации выполняется по методу Оцу. Так как после предыдущего шага на изображении присутствует как бы два класса пикселей – фоновые и объектные, Оцу подходит для определения границы бинаризации лучше – выше будет межклассовая дисперсия [ImageThresholding, 2018].
4. Определение контуров объектов на бинарном изображении: далее по списку – определение контуров объектов на бинаризованном изображении.В OpenCV используется алгоритм топологического структурного анализа бинарных изображений, предложенный Сатоши Сузуки и Кейчи Эйбом [Satoshi, Keiichi, 1985]. Алгоритм предполагает нахождение контуров с учетом вложенности, то есть способен определить, когда в контур одного объекта вложен другой. В

предметной области исследования это может возникать при наличии на снимке здоровых двояковогнутых эритроцитов. При этом при засветке вогнутость эритроцита не фиксируется матрицей камеры, таким образом, объект получается с «дыркой», и с точки зрения алгоритма Suzuki85 содержит два объекта, а полную площадь можно посчитать, сложив площади самого объекта и его «дырки».В OpenCV данный режим для cv::findContours называется CV_RETR_CCOMP. Он извлекает все контуры и организует их в двухуровневую иерархию. На верхнем уровне существуют внешние границы компонентов. На втором уровне есть границы отверстий. Если в отверстии подключенного компонента есть еще один контур, он все еще находится на верхнем уровне.

5. Выделение каждой клетки в свой класс - классификация областей: пиксели, которые соответствуют номеру класса, красятся в соответствующий цвет.Далее для наглядности и удобства расчета площади объекта, найденные контуры заливаются случайными неповторяющимися цветами.Алгоритм работает с изображением как с функцией от двух переменных f = I(x,y), где x,y – координаты пикселя.Значением функции может быть интенсивность или модуль градиента. Для наибольшего контраста можно взять градиент от изображения. Если по оси O_z откладывать абсолютное значение градиента, то в местах перепада интенсивности образуются хребты, а в однородных регионах – равнины. После нахождения минимумов функции f, идет процесс заполнения “водой”, который начинается с глобального минимума. Как только уровень воды достигает значения очередного локального минимума, начинается его заполнение водой. Когда два региона начинают сливаться, строится перегородка, чтобы предотвратить объединение областей. Вода продолжит подниматься до тех пор, пока регионы не будут отделяться только искусственно

построенными перегородками [Watershedapproachesforcolorimagesegmentation, 2018; Beucher, Meyer, 1992].В данном случае, информацией о перегородках выступают контуры объектов с предыдущего шага, а значение пикселей на контурах – высотой плато. Таким образом, каждая равнина (группа пикселей, заключенных в контур) является уникальным классом. За счет иерархичности контуров, правильно (в рамках предметной области, то есть не артефакты и ошибки сегментации) заливают и вложенные объекты.

2 ТЕОРЕТИЧЕСКИЕ ОСНОВЫ

2.1 Алгоритм гематологического анализа клеток крови

Рассмотрим автоматизированный гематологический анализ клеток крови. Технология автоматического гематологического анализа предполагает подсчет клеток и их геометрических характеристик. Точность вычислительного эксперимента очень важна, так как затрагивается здоровье человека, и тут главную роль играют математические основы компьютерной обработки изображений. Выделим главные задачи математики в компьютерной обработке изображений:

- с помощью математических фильтров происходит предварительная обработка изображения;
- математические виды пороговой сегментации классифицируют изображение на области клеток крови (эритроцитов);
- строятся наиболее точные границы областей (с применением математических фильтров);
- для каждого объекта (клеток крови) считается площадь; – Вычисляется диаметр каждой области на основе площадей;
- строится кривая Прайс-Джонса.

Основным результатом сегментации микроскопического изображения мазка крови является выделение объектов интереса (клеток крови) с целью их дальнейшей классификации. Качество сегментации – ключевой фактор для получения адекватных значений характерных признаков объекта, а значит, для его верной классификации. Так, в работе [4] отмечено, что более половины ошибок классификации лейкоцитов были обусловлены неправильной сегментацией.

Основные причины, ведущие к ошибкам сегментации микроскопических изображений: перекрывание одной клетки другой, сильная вариация клеток по форме и размеру, воздействие разных факторов на внешний вид клетки, слабая контрастность изображений с дополнительными проблемами, вызываемыми шумами. Еще один усложняющий фактор – вариабельность окраски препаратов крови: часто после окраски контрастность контуров структурных элементов внутри клетки превышает контрастность границ самой клетки.

Предлагаемый нами [5] метод предварительной обработки изображений предусматривает следующие этапы:

1. смена цветового пространства;
2. удаление заднего фона;
3. бинаризация изображения;
4. определение контуров;
5. классификация областей.

2.2 Смена цветового пространства

Перевод изображения в цветовое пространство HSL (HLS в представлении OpenCV) с выделением отдельно канала L – светимость конкретного пикселя. Специфика исходных данных (изначально изображения представлены в оттенках серого) позволяет первым этапом выделить один канал и с ним дальше работать. Также, большинство методов OpenCV, необходимых для обработки изображения, тоже работают с одним каналом. В этом есть и доля оптимизации – нет необходимости на каждом этапе L-канал из изображения и в конце восстанавливать канальность изображения в исходное состояние.

В общем виде перевод цветового пространства изображения из RGB в HSL можно записать как:

$$V_{max} \leftarrow \max(R, G, B) \tag{2.1}$$

$$V_{min} \leftarrow \min(R, G, B) \tag{2.2}$$

$$L \leftarrow \frac{V_{max} + V_{min}}{2} \tag{2.3}$$

$$S \leftarrow \begin{cases} \dfrac{V_{max} - V_{min}}{V_{max} + V_{min}}, L < 0.5 \\ \dfrac{V_{max} - V_{min}}{2 - (V_{max} + V_{min})}, L \geq 0.5 \end{cases} \tag{2.4}$$

$$SH \leftarrow \begin{cases} \dfrac{60(G - B)}{S}, V_{max} = R \\ 120 + \dfrac{60(B - R)}{S}, V_{max} = G \\ 240 + \dfrac{60(R - G)}{S}, V_{max} = B \end{cases} \tag{2.5}$$

Но так как в изображении в оттенках серого каждый пиксель в RGB кодируется одинаковыми значениями для каждой компоненты, достаточно выделить один любой канал в качестве L-канала HSL. Однако, правильное разложение по каналам оставлено на тот случай, если выходные данные будут представляться полноцветными изображениями.

2.3 Удаление заднего фона

Удаление заднего фона на изображениях: в некоторых местах есть области пересвечивания пикселей и наоборот – затемнения, из-за которых происходит неправильное контурирование клетки. Так как изначально изображение зашумлено, необходимо также сгладить перепады между группой соседних пикселей.

Сглаживание изображения просто сделать через размытие по Гауссу. Ядра размером 3x3 вполне достаточно, чтобы убрать неравномерности в шумах фона, но в то же время не потерять границы объектов. Пример сглаживания представлен на рис. 2.2 Исходное изображение представлено на рис. 2.1.

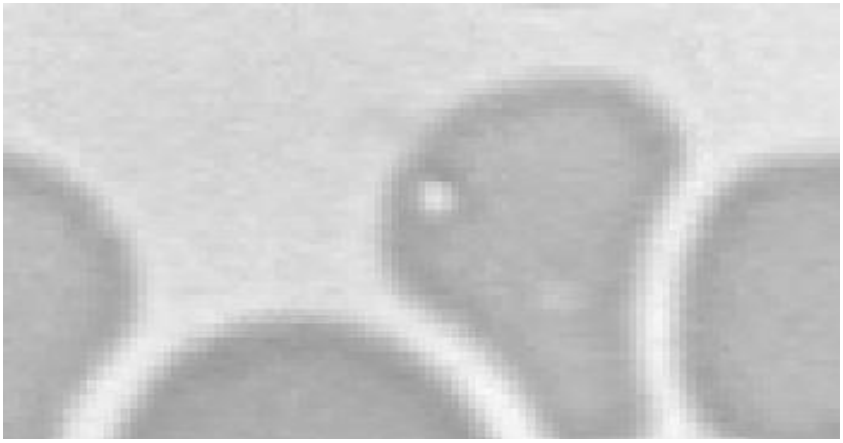

Рис. 2.1. Исходное изображение

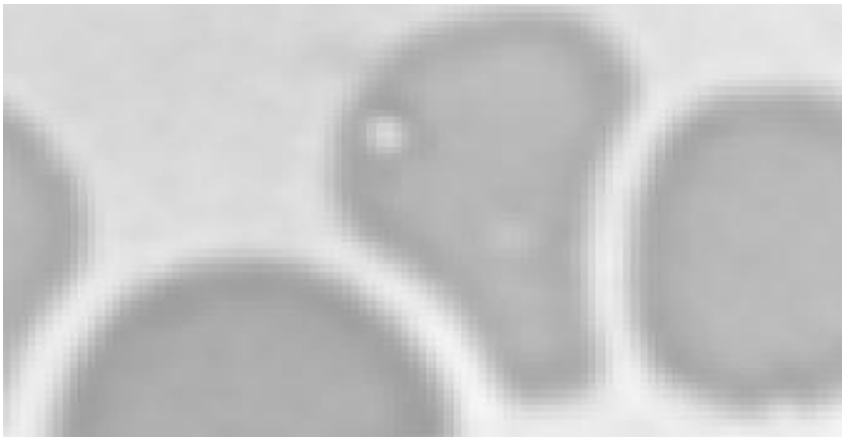

Рис.2.2. Применение размытия Гаусса

После выравнивания фоновых пикселей изображения необходимо увеличить контрастность изображения, чтобы при удалении фоновых пикселей оставить пиксели объектов, которые примерно равны по светимости фоновым.

Для увеличения контрастности изображения используется алгоритм CLAHE (Contrast Limited Adaptive Histogram Equalization) [7]. Данный алгоритм, как и все алгоритмы с выравниванием гистограммы, использует функцию плотности вероятности (2) и кумулятивную функцию плотности (3) для приведения гистограммы интенсивности светимости пикселей к нужному виду. Если принять, что N – количество пикселей в изображении, L – общее количество оттенков серого (уровней интенсивности) на изображении, а n_k – это общее количество пикселей со светимостью l_i, то функция плотности вероятности и кумулятивная функция плотности будут иметь вид:

$$f_i(i_k) = \frac{n_k}{N} \tag{2.6}$$

$$F_k(i_k) = \sum_{j=0}^{k} f_i(i_j) \tag{2.7}$$

Отличие CLAHE от других алгоритмов в том, что он выполняет эквализацию на ограниченной области изображения с предварительно ограниченной гистограммой светимости. Таким образом, он более устойчив к порождению шума на гомогенных областях.

Результат применения алгоритма CLAHE на исследуемом изображении приведен на рис. 2.3.

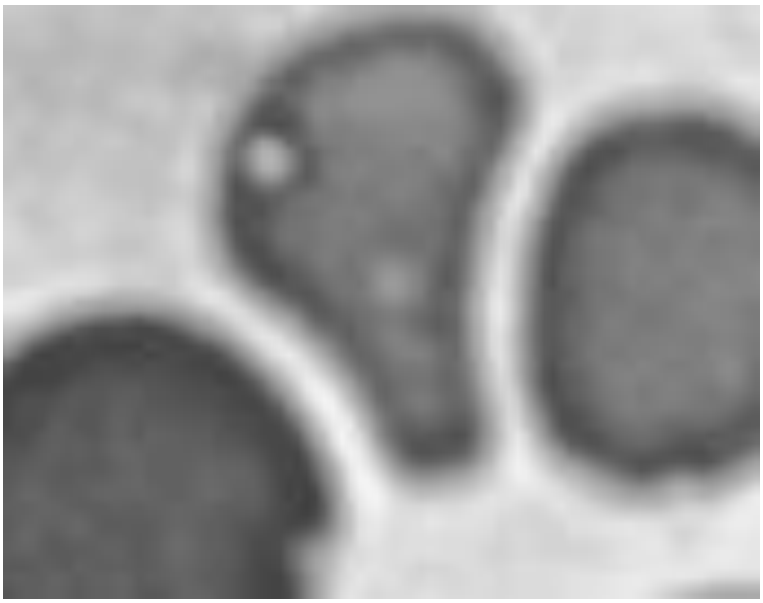

Рис.2.3. Применения алгоритма CLAHE

Теперь, когда клетки (объекты) сильно контрастируют с фоном, можно удалить фоновые пиксели с изображения.

$$x_i = \begin{cases} 255, x_i \geq \text{mean}(X) * 0.9 \\ x_i, \text{в остальных случаях'} \end{cases} \quad (2.8)$$

где x_i – светимость i-го пикселя изображения, X – множество всех пикселей.

То есть, в том случае, если светимость пикселя больше или равна 90% средней светимости по всему изображению, ему присваивается значение 255 – белый цвет, остальные пиксели неизменны. Значение в 90% было подобрано эмпирически. При таком пороге удаляются практически все фоновые пиксели, при этом остаются неизменными пиксели объектов. Пример удаления фона приведен на рис. 2.4.

Рис.2.4. Результат удаления фоновых пикселей

2.4 Бинаризация изображения

Следующим шагом необходимо выполнить бинаризацию изображения, то есть приведение изображение к виду, когда каждый пиксель кодируется либо единицей, либо нулем. Этот шаг необходим так как некоторые последующие функции OpenCV используют бинарное изображение, как один из аргументов.

Выбор порога сегментации выполняется по методу Оцу. Так как после предыдущего шага на изображении присутствует как бы два класса пикселей – фоновые и объектные, Оцу подходит для определения границы бинаризации лучше – выше будет межклассовая дисперсия. Пример бинаризации изображения приведен на рис.2.5.

Рис.2.5. Бинаризация изображения

2.5 Определение контуров объектов на бинарном изображении

Далее по списку – определение контуров объектов на бинаризованном изображении.

В OpenCVиспользуется алгоритм топологического структурного анализа бинарных изображений, предложенный Сатоши Сузуки и Кейчи Эйбом[8].Алгоритм предполагает нахождение контуров с учетом вложенности, то есть способен определить, когда в контур одного объекта вложен другой. В предметной области исследования это может возникать при наличии на снимке здоровых двояковогнутых эритроцитов. При этом при засветке вогнутость эритроцита не фиксируется матрицей камеры, таким образом, объект получается с «дыркой», и с точки зрения алгоритма Suzuki85 содержит два объекта, а полную площадь можно посчитать, сложив площади самого объекта и его «дырки».

В OpenCVданный режим для cv::findContoursназывается CV_RETR_CCOMP. Он извлекает все контуры и организует их в двухуровневую иерархию. На верхнем уровне существуют внешние границы

компонентов. На втором уровне есть границы отверстий. Если в отверстии подключенного компонента есть еще один контур, он все еще находится на верхнем уровне.

Пример выделения контуров объектов представлен на рис.2.6.

Рис.2.6. Определение контуров объектов

2.6 Выделение каждой клетки в свой класс

Классификация областей: пиксели, которые соответствуют номеру класса, красятся в соответствующий цвет.

Далее для наглядности и удобства расчета площади объекта, найденные контуры заливаются случайными неповторяющимися цветами.

Алгоритм работает с изображением как с функцией от двух переменных $f = I(x, y)$, где x, y – координаты пикселя.

Значением функции может быть интенсивность или модуль градиента. Для наибольшего контраста можно взять градиент от изображения. Если по оси O_z откладывать абсолютное значение градиента, то в местах перепада интенсивности образуются хребты, а в однородных регионах – равнины. После нахождения минимумов функции f, идет процесс заполнения “водой”,

который начинается с глобального минимума. Как только уровень воды достигает значения очередного локального минимума, начинается его заполнение водой. Когда два региона начинают сливаться, строится перегородка, чтобы предотвратить объединение областей. Вода продолжит подниматься до тех пор, пока регионы не будут отделяться только искусственно построенными перегородками.

В данном случае, информацией о перегородках выступают контуры объектов с предыдущего шага, а значение пикселей на контурах – высотой плато. Таким образом, каждая равнина (группа пикселей, заключенных в контур) является уникальным классом. За счет иерархичности контуров, правильно (в рамках предметной области, то есть не артефакты и ошибки сегментации) заливают и вложенные объекты.

3 РАЗРАБОТКА ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

3.1 Структура программы

Разработанному программному обеспечению на вход поступает изображение, полученное с помощью оптического микроскопа. Далее полученное изображение обрабатывается последовательно, исходя из предложенного алгоритма. На выходе получаем обработанное изображение, с посчитанными площадями каждой клетки. Блок-схема программы представлена на рис. 3.1.

Рис.3.1. Блок-схема

3.2 Описание реализации программы

Первым этапом обработки исходного изображения является перевод изображения в цветовое пространство HSL (HLS в представлении OpenCV) с выделением отдельно канала L – светимость конкретного пикселя. Специфика исходных данных (изначально изображения представлены в оттенках серого) позволяет первым этапом выделить один канал и с ним дальше работать. Также, большинство методов OpenCV, необходимых для обработки изображения, тоже работают с одним каналом. В этом есть и доля оптимизации – нет необходимости на каждом этапе L-канал из изображения и в конце восстанавливать канальность изображения в исходное состояние.

Перевод изображения в цветовое пространство HSL реализован в листинге 3.1.

Листинг3.1. Смена цветового пространства

```
classBGR2YUV(BaseLayer):
defprocess(self):
self._output = cv2.cvtColor(self.input, cv2.COLOR_BGR2YUV)
```

Конец листинга.

Вторымэтапомявляетсяудаление заднего фона. Для этого сначала необходимо увеличить контрастность изображения, а затем удалить фон методом `threshold`.Дляувеличенияконтрастностиизображенияиспользуетсяалгоритм CLAHE (ContrastLimitedAdaptiveHistogramEqualization) [Sasi, Jayasree, 2013]. Данный алгоритм, как и все алгоритмы с выравниванием гистограммы, использует функцию плотности вероятности (1) и кумулятивную функцию плотности (2) для приведения гистограммы интенсивности светимости пикселей к нужному виду.

Применение алгоритмаCLAHE показано на листинге 3.2.

Листинг 3.2. АлгоритмCLAHE

```
class CLAHE(LumaLayer):
    def __init__(self, **kwargs):
        self.CLAHE = None
        super(CLAHE, self).__init__(**kwargs)

    def configure(self):
        self.param("clipLimit", 3.0)
        self.param("tileGridSize", (7, 7))

    def prepare(self):
        self.CLAHE =
cv2.createCLAHE(clipLimit=self.params["clipLimit"],

tileGridSize=self.params["tileGridSize"])

    def process(self):
        self._output = self.CLAHE.apply(self._input)
```

Конец листинга.

Теперь, когда клетки (объекты) сильно контрастируют с фоном, можно удалить фоновые пиксели с изображения

Удаление фоновых пикселей представлено на листинге 3.3.

Далее определяем центры клеток. Получаем изображение, которое представляет собой бинарное изображение, где значение каждого пикселя заменяется его расстоянием до ближайшего пикселя фона.

Чтобы получить производное представление двоичного изображения, где значение каждого пикселя заменяется его расстоянием до ближайшего фонового пикселя используется функция`DistanceTransform`.

Листинг 3.3. Удалениефона

```
     class Threshold(LumaLayer):
    def configure(self):
        self.param("thresh", self._input.mean())
        self.param("maxval", 255)
        self.param("type", cv2.THRESH_OTSU |
cv2.THRESH_BINARY_INV)

    def process(self):
        _, self._output = cv2.threshold(self._input,
                                        self.params["thresh"],
                                        self.params["maxval"],
                                        self.params["type"])\
```

Конецлистинга.

Листинг 3.4. Бинаризацияизображения

```
classDistanceTransform(LumaLayer):

defconfigure(self):
self.param("type", cv2.DIST_L2)
self.param("maskSize", 3)

defprocess(self):
self._output = cv2.distanceTransform(self._input,
self.params["type"],
self.params["maskSize"])
```

Конец листинга.

Далее по списку – определение контуров объектов на бинаризованном изображении.В OpenCVданный режим для cv::findContoursназывается CV_RETR_CCOMP. Он извлекает все контуры и организует их в двухуровневую иерархию. На верхнем уровне существуют внешние границы

компонентов. На втором уровне есть границы отверстий. Если в отверстии подключенного компонента есть еще один контур, он все еще находится на верхнем уровне.

Листинг 3.5. Поискконтуров

```
class FindContours(LumaLayer):
    # TODO: set (0,0) pixel to background color, thus Suzuki85
algo should work fine and find left-top-most object

def configure(self):
        self.param("mode", cv2.RETR_EXTERNAL)  #
cv2.RETR_EXTERNAL
        self.param("method", cv2.CHAIN_APPROX_SIMPLE)

    def prepare(self):
        self._input = np.uint8(self._input.copy())

    def process(self):
        image, contours, hierarchy =
cv2.findContours(self._input,

self.params["mode"],

self.params["method"])

        self._output = {"image": image,
                        "contours": contours,
                        "hierarchy": hierarchy}
```

Конец листинга.

Контуры можно объяснить просто как кривая, соединяющая все непрерывные точки (вдоль границы), имеющие одинаковый цвет или

интенсивность.Контуры являются полезным инструментом для анализа формы и обнаружения, и распознавания объектов.

Для рисования контуров используется функция cv2.drawContours. Его также можно использовать для рисования любой формы, если у вас есть ее граничные точки. Его первый аргумент - исходное изображение, второй аргумент - контуры, которые должны быть переданы как список Python, третий аргумент - индекс контуров (полезно при рисовании индивидуального контура. Чтобы нарисовать все контуры, передать -1), а остальные аргументы - цвет, толщина и т.п. Рисование контуров методами OpenCV представлено на листинге 3.6.

Листинг 3.6 Прорисовкаконтуров

```
 class DrawContours(BaseLayer):
    def validateInput(self):
        assert "image" in self._input
        assert "contours" in self._input
        #assert "hierarchy" in self._input
    def prepare(self):
        self.image = self._input["image"]
        self.contours = self._input["contours"]
        #self.hierarchy = self._input["hierarchy"]
        self.markers = np.zeros(self.image.shape,
dtype=np.int32)
    def process(self):

        for idx in xrange(1, len(self.contours)):
            cv2.drawContours(self.markers, self.contours, idx -
1, idx)

        self._output = self.markers
```

Конец листинга.

Классификация областей: пиксели, которые соответствуют номеру класса, красятся в соответствующий цвет.

OpenCV реализовал алгоритм водоразделов на основе маркеров, в котором вы указываете, какие точки долины должны быть объединены, а какие нет. Это интерактивная сегментация изображений. Мы делаем это, чтобы дать разные метки для нашего объекта, который мы знаем. Назовите область, в которой мы уверены, что являемся передним планом или объектом с одним цветом (или интенсивностью), назовите область, в которой мы уверены, являемся фоном или не объектом другого цвета и, наконец, область, в которой мы не уверены ни в чем, наклейте его на 0. Это наш маркер. Затем примените алгоритм водораздела. Затем наш маркер будет обновлен с метками, которые мы дали, а границы объектов будут иметь значение -1.

Листинг 3.7. Алгоритмводораздела

```
 class Watershed(LumaLayer):
    def validateInput(self):
        assert "srcimage" in self._input
        assert "image" in self._input
assert "markers" in self._input
    def configure(self):
        self.param("basecolor", (255, 255, 255))
    def prepare(self):
        self.srcimage = self.input["srcimage"]
        self.image = self.input["image"]
        self.markers = self.input["markers"]
        self.colors = np.int32(np.random.randint(100, 254,
size=(len(self.markers), 3)))
    def process(self):
        cv2.watershed(self.srcimage, self.markers)
        for i in xrange(0, self.markers.max()):
            self.srcimage[self.markers == i] = self.colors[i]
```

Конец листинга.

После предварительной обработки изображения легко считаются площади клеток крови по наиболее тонким и четким контурам. Из площади клеток крови вычисляем диаметр каждой клетки и строим кривую Прайс-Джонса, которая представляет себе гистограмму распределе ния диаметров клеток, где по оси абсцисс откладывают величину диаметра эритроцитов (в мкм), а по оси ординат – процентное содержание лейкоцитов соответствующей величины.

Подсчет площади, нахождения диаметров и вывод кривой Прайс-Джонса показан на листинге 3.8.

Листинг3.8. Вывод кривой Прайс-Джонса

```
    pipeline.input = imgFileName
pipeline.run()
squares = pipeline.layers["csquares"].output
diams = [0.154 * 0.154 * x / 3.14 forxinsquares]
seaborn.distplot(diams, hist=True, kde=True)
plt.ylabel(u"Плотностьвероятности")
plt.xlabel(u"Диаметр(мкм)")
plt.show()
cv2.imshow("out", pipeline.layers["watersh"].output)
cv2.waitKey(0)
```

Конец листинга.

4 ВЫЧИСЛИТЕЛЬНЫЙ ЭКСПЕРИМЕНТ

Разработанное программное обеспечение позволят проводить сложный статический анализ заданной выборки клеток и на основе этого построить кривую Прайс-Джонса. Для запуска программы необходимо отсканировать мазок крови и тем самым получить исходное изображение для дальнейшей обработки.

Первым этапом является смена цветового пространства, это делается затем, что методы OpenCV, используемые для обработки изображений, работают с одним каналом. Исходное изображение, полученное с микроскопа представлено на рис. 4.1.

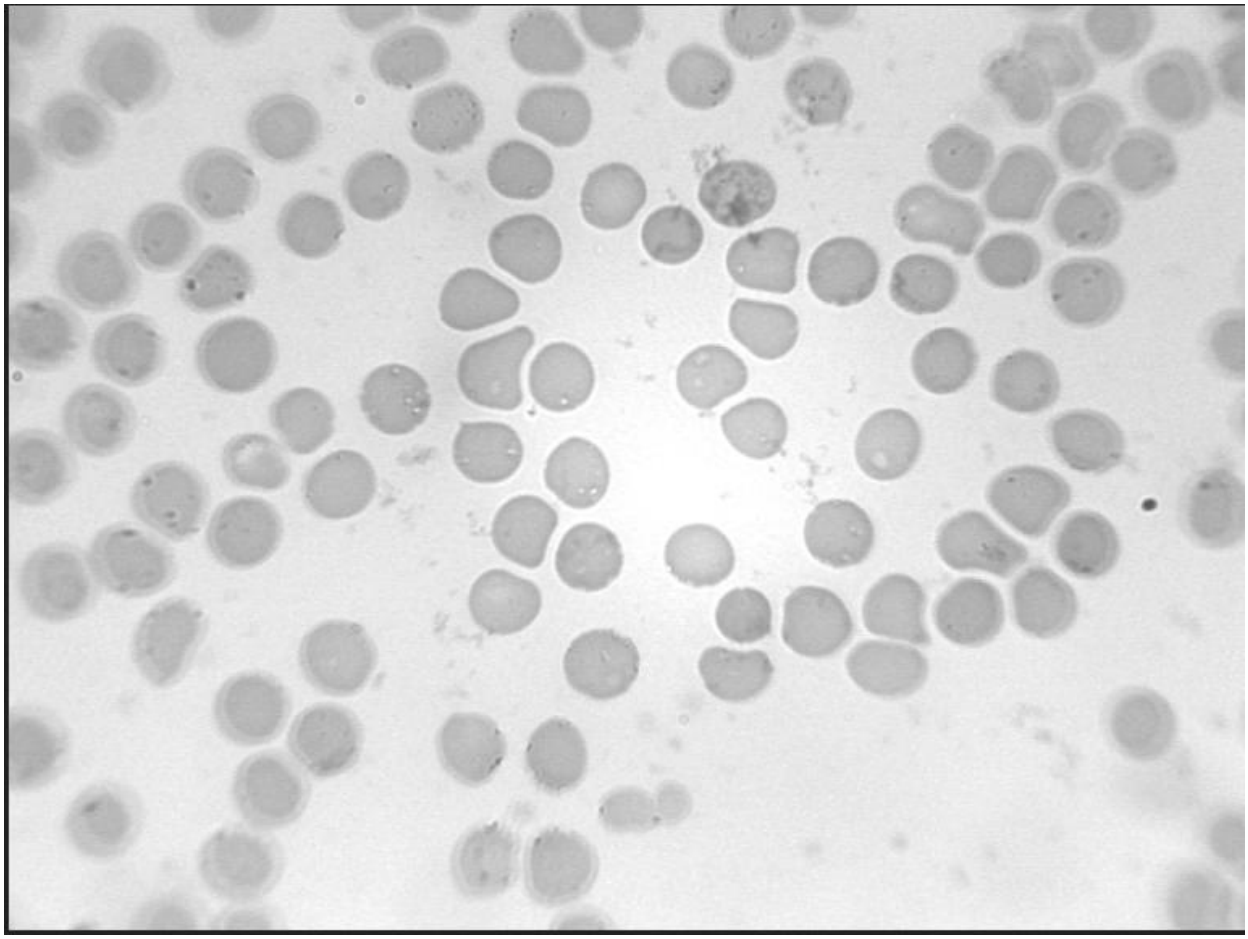

Рис.4.1. Исходное изображение в пространстве RGB

Изображение в HSL представлено на рис. 4.2.

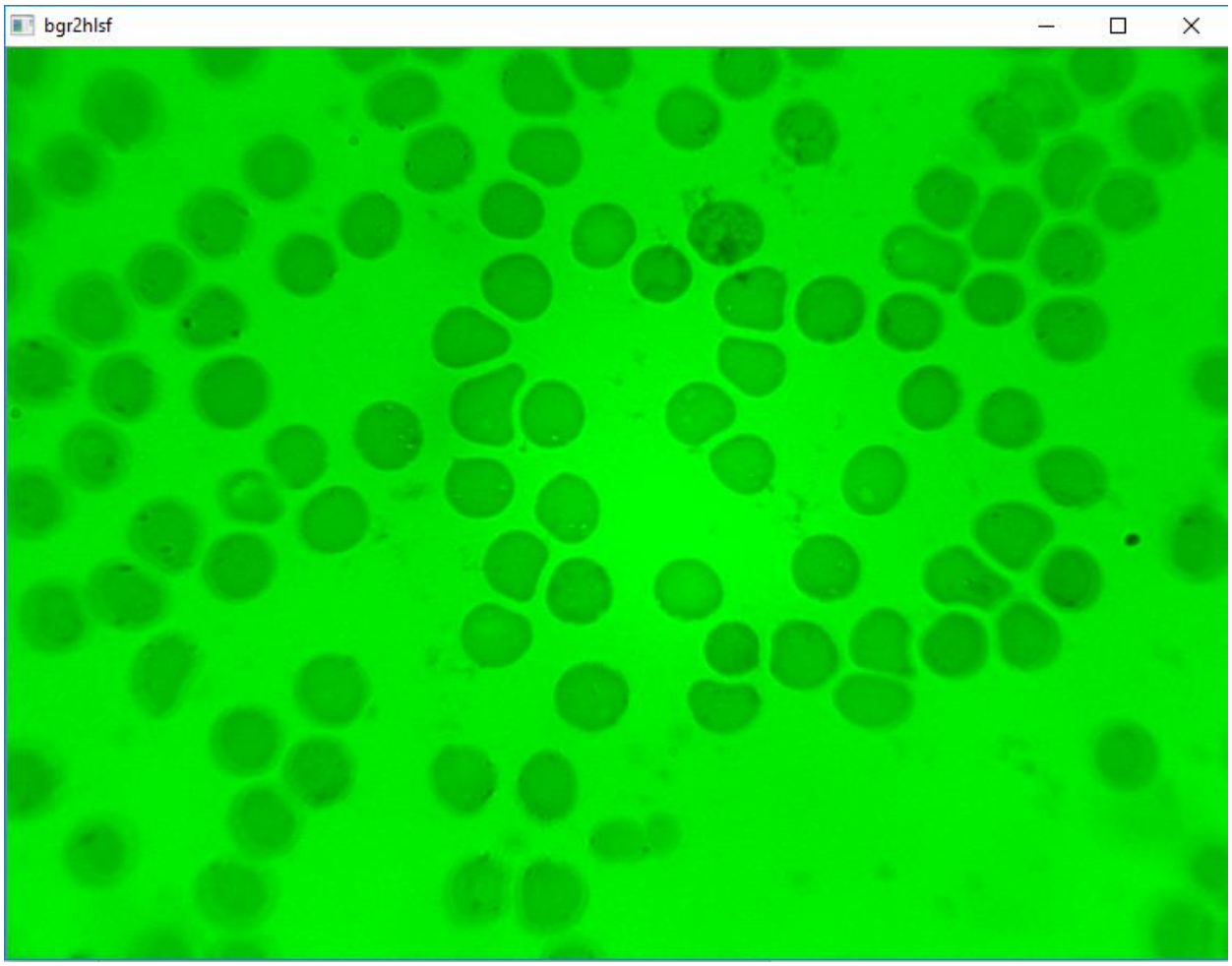

Рис.4.2. Исходное изображение в пространстве HSL

Изображение изначально зашумлено, поэтому его необходимо сгладить фильтром Гаусса. Фильтр «Гауссово размывание» меняет каждую точку текущего слоя или выделения, делая её значение равным среднему значению всех точек в определённом радиусе от рассматриваемой точки. Значение этого радиуса можно изменить. Чем больше радиус, тем сильнее будет размыто изображение. Размывание можно усилить в одном направлении по сравнению с другим, разорвав зависимость между радиусом по горизонтали и вертикали. GIMP поддерживает два алгоритма фильтра: IIR и RLE. Эти алгоритмы дают одинаковый результат, но в определённых случаях один

может быть быстрее другого Изображение после применения фильтра Гаусса представлено на рис. 4.3.

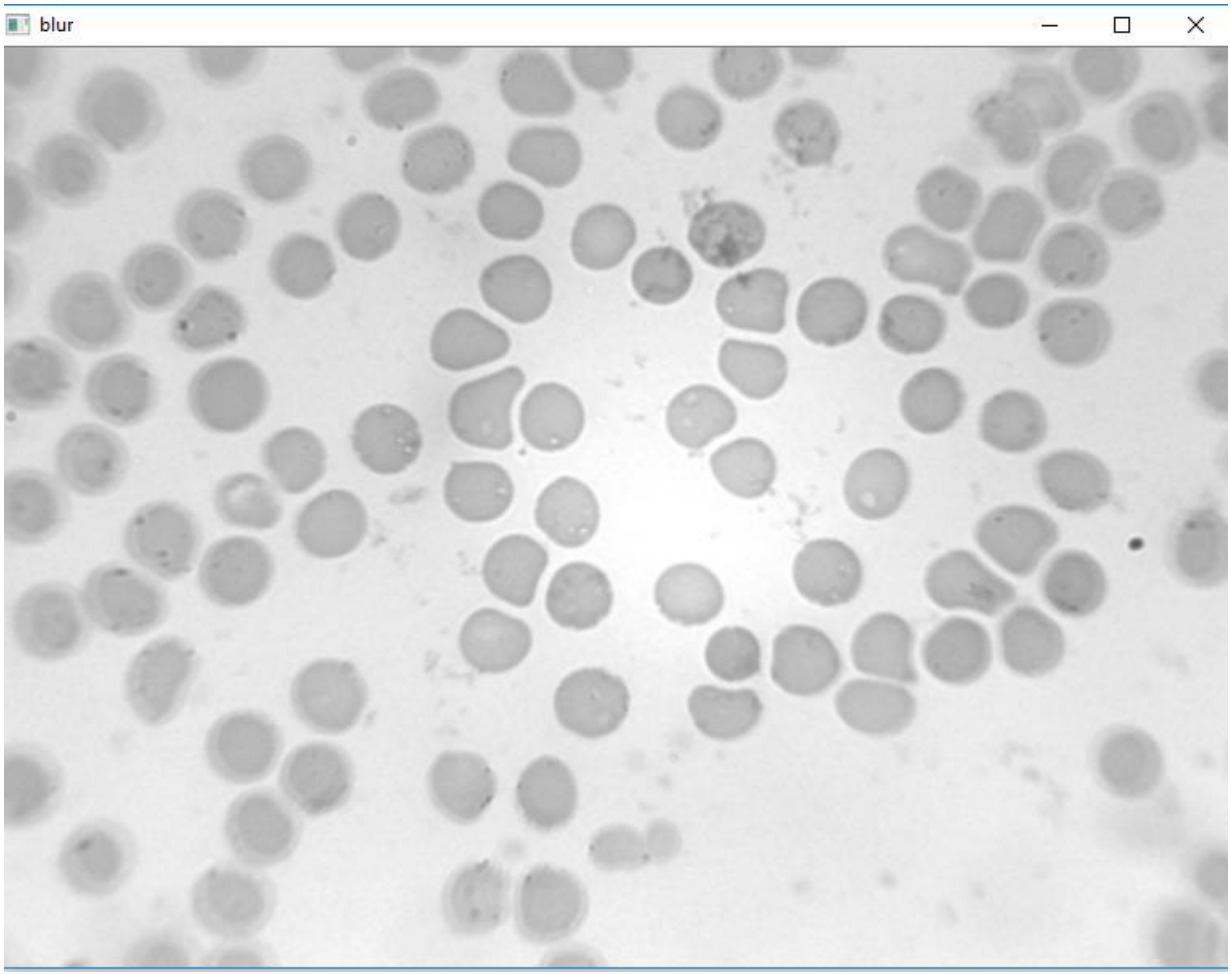

Рис.4.3. Изображение после обработки фильтром Гаусса

Увеличение контрастности изображения показано на рис. 4.4.Контраст представляет собой характеристику того, насколько большой разброс имеют цвета пикселей изображения. Чем больший разброс имеют значения цветов пикселей, тем больший контраст имеет изображение.

После увеличения контрастности изображения необходимо удалить задний фон. Выполняется фиксированное пороговое преобразование для элементов массива. Изображение после удаления заднего фона представлено на рис. 4.5.

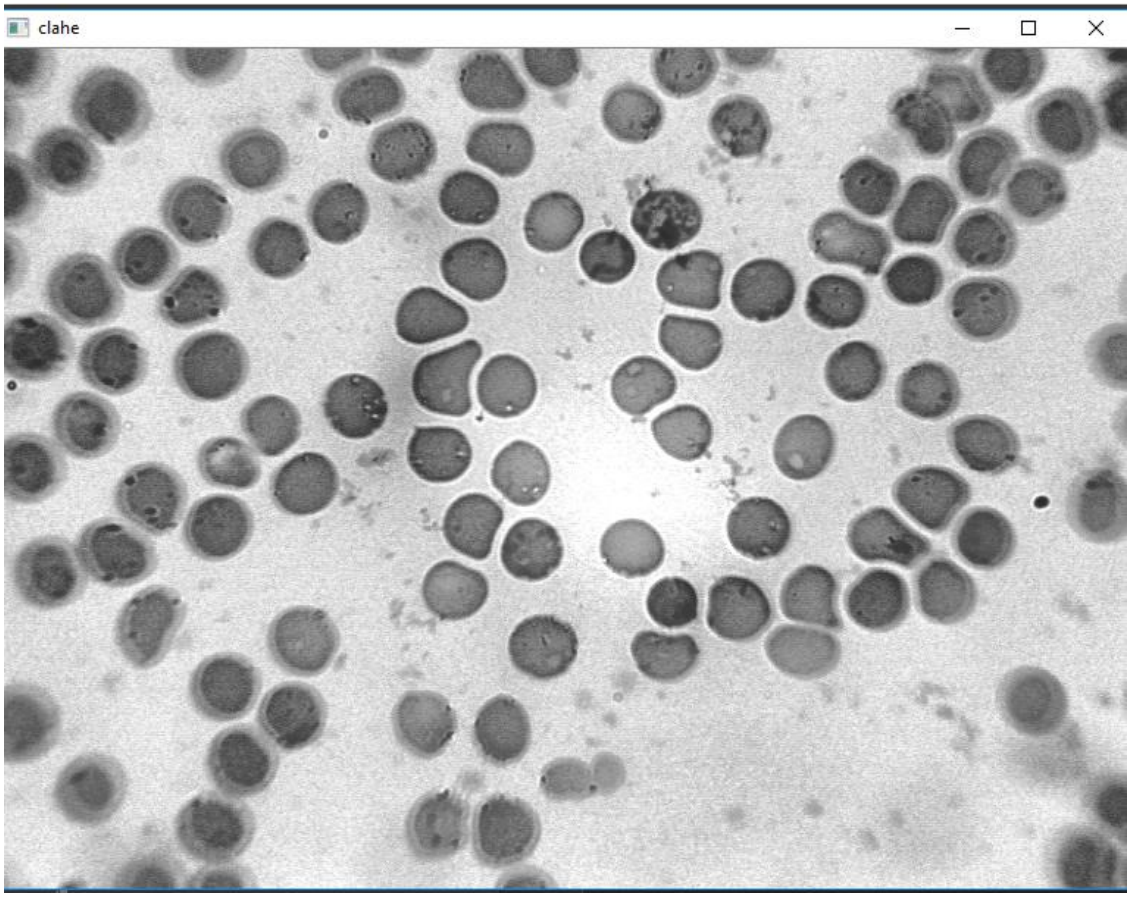

Рис.4.4. Изображение после увеличения контрастности

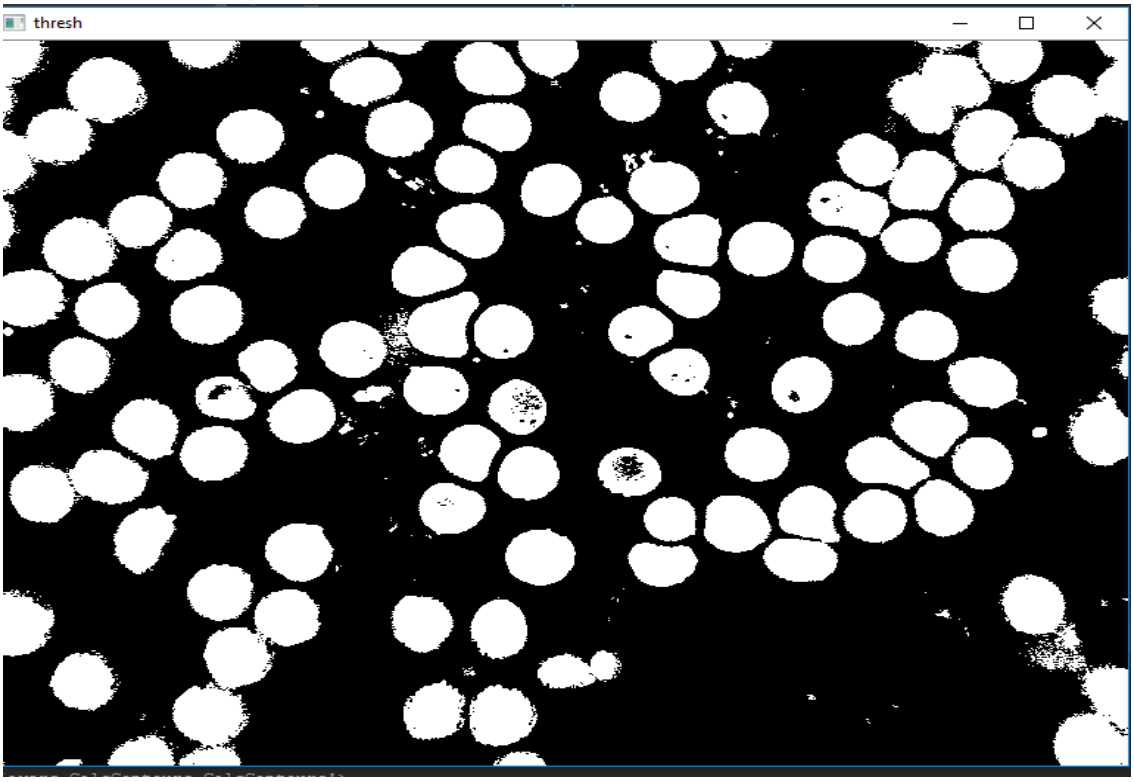

Рис.4.5. Изображение после удаления заднего фона

Далее необходимо найти цент каждой клетки. Изображение с центрами клеток представлено на рис. 4.6.

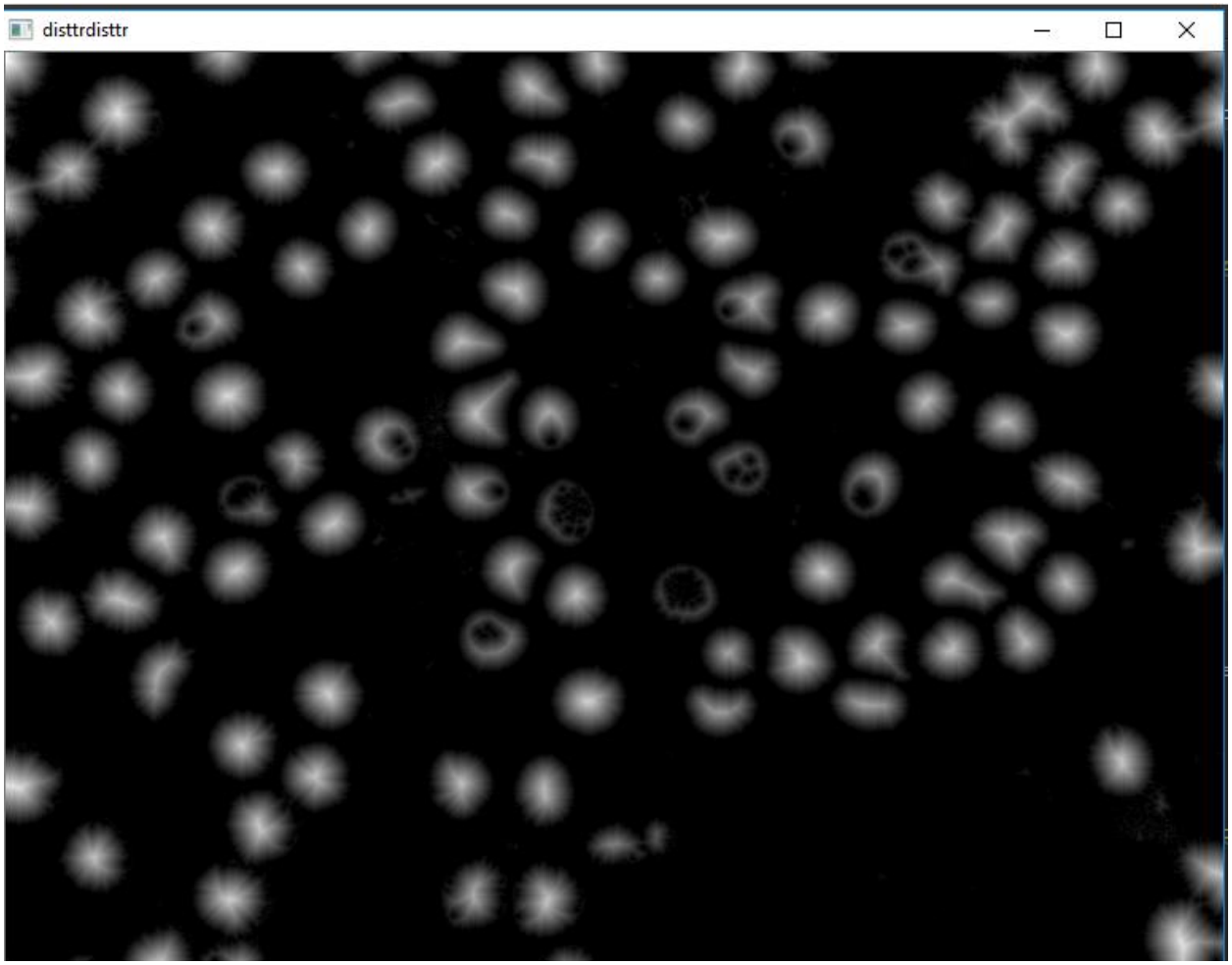

Рис.4.6. Изображение с центрами клеток

Далее находим контуры клеток и рисуем их. Для улучшения точности подсчета площадей клеток крови необходимо, чтобы контуры были наиболее тонкими. Это позволит правильно подсчитать параметры снимков клеток крови и дать наиболее точный результат. Изображение с выделенными контурами представлено на рис. 4.7.

Последним этапом является классификация клеток. Для наглядности и удобства расчета площади объекта, найденные контуры заливаются случайными неповторяющимися цветами. Полученное изображение представлено на рис. 4.8.

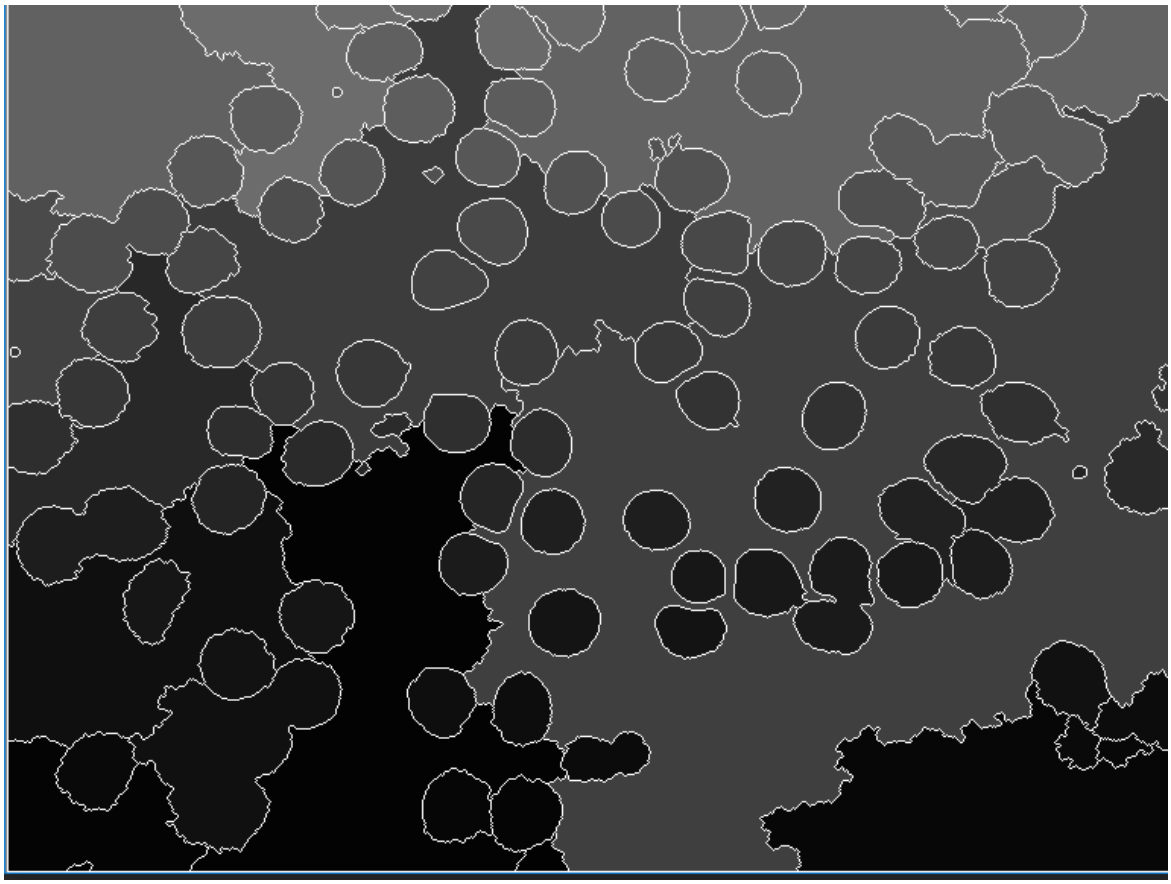

Рис.4.7. Отрисованные контуры каждой клетки

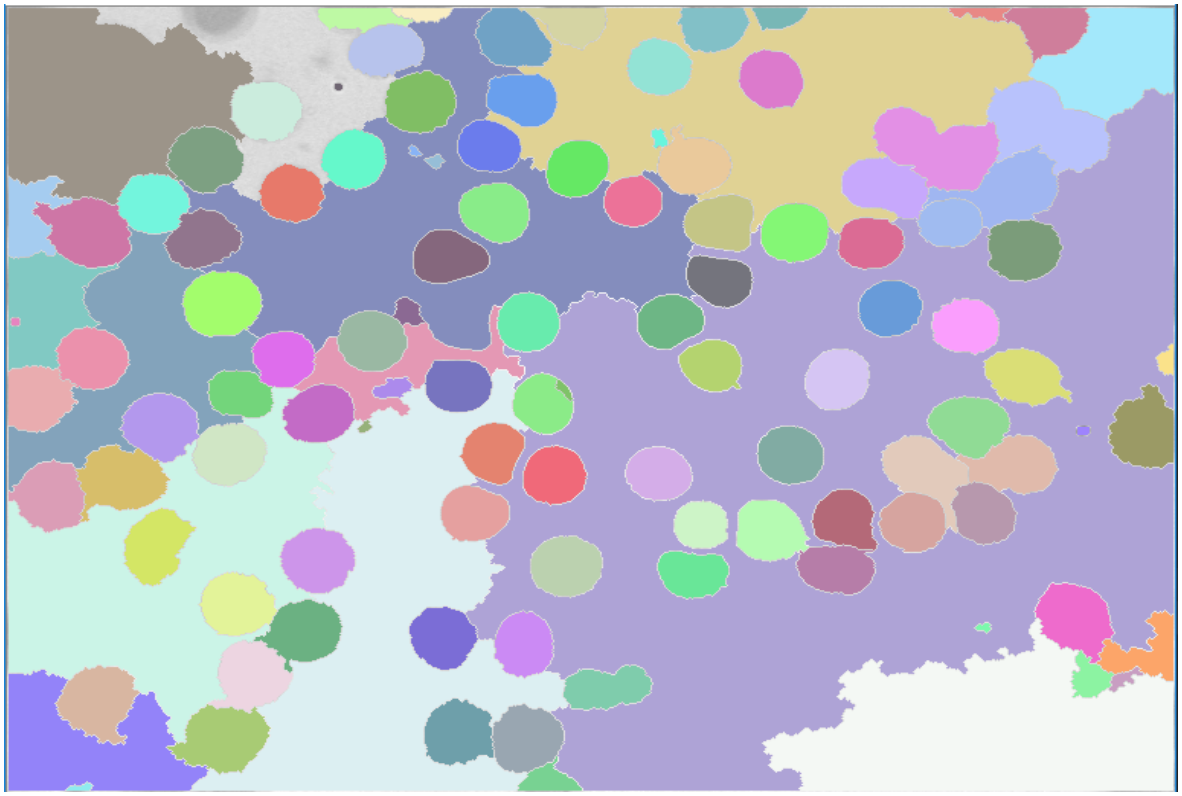

Рис.4.8. Классификация клеток

На данном этапе найдены все атрибуты для подсчета площадей каждой клетки. Считаем площадь каждой клетки, находим распределение площадей клеток по их диаметру и строим кривую Прайс-Джонса.

Из площади клеток крови вычисляем диаметр каждой клетки и строим кривую Прайс-Джонса, которая представляет себе гистограмму распределения диаметров клеток, где по оси абсцисс откладывают величину диаметра эритроцитов (в мкм), а по оси ординат – процентное содержание лейкоцитов соответствующей величины.Гистограмма распределения диаметров показана на рис. 4.9.

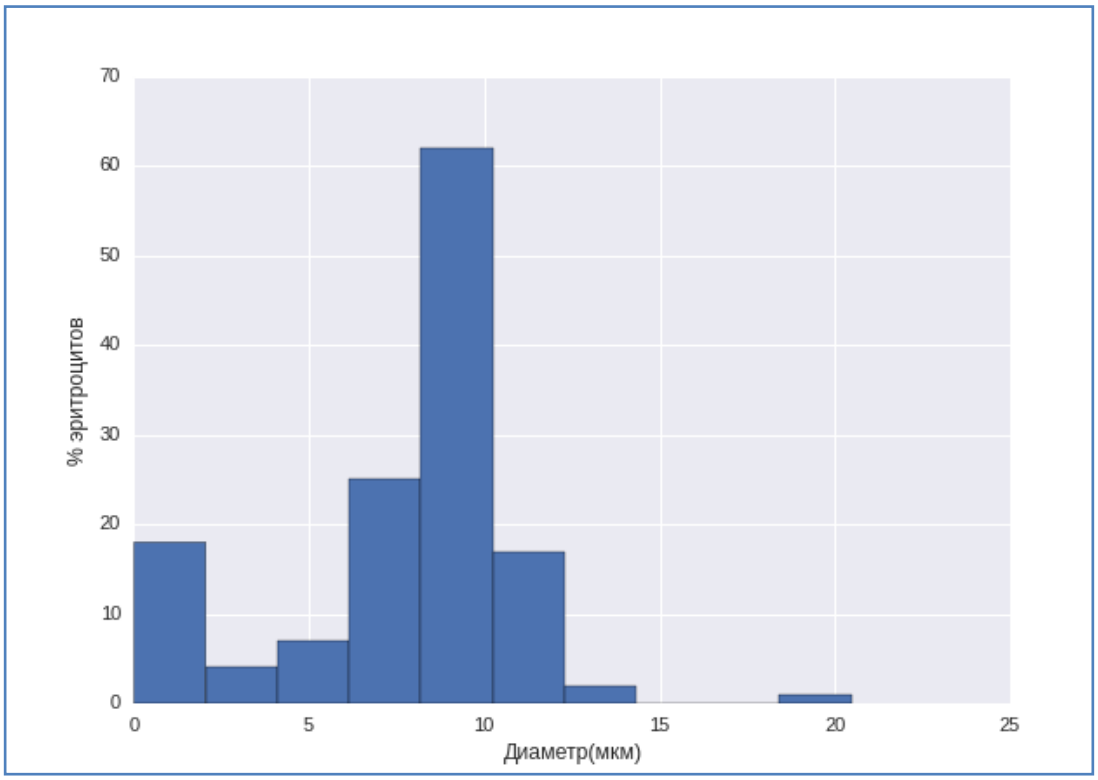

Рис.4.9. Гистограмма распределения диаметров

Кривая Прайс-Джонса больного человека показана на рис. 4.11. Это видно из-за того, что кривая немного сдвинута влево.

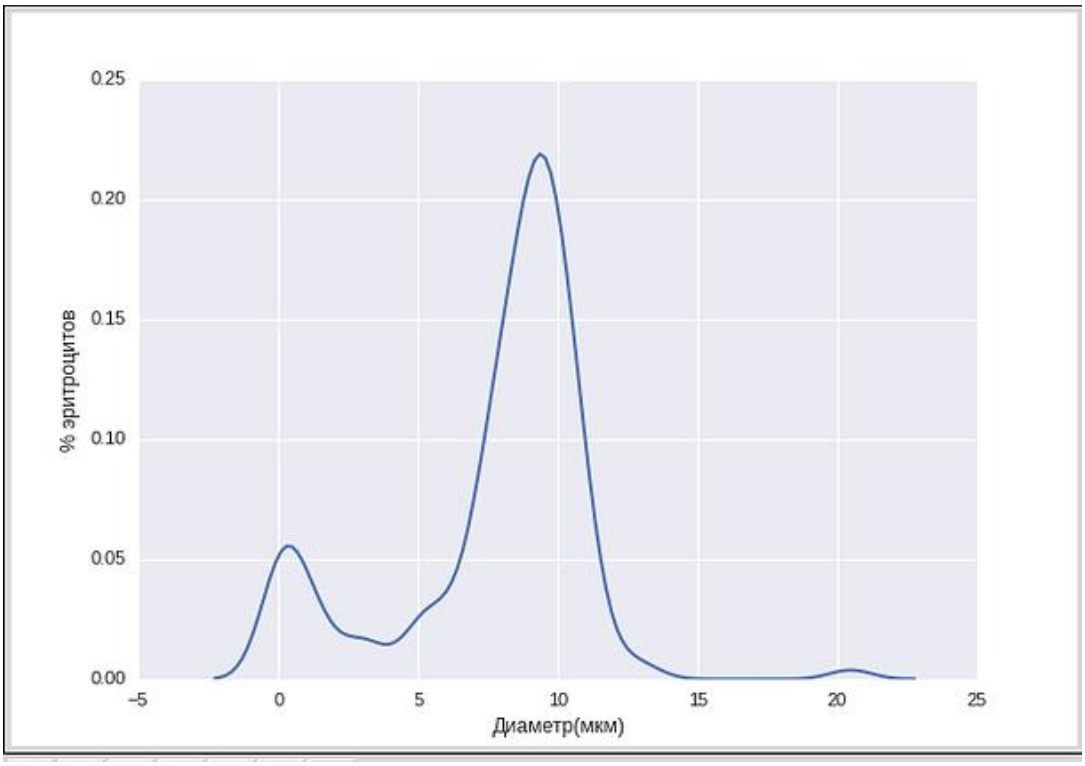

Рис.4.10. Кривая Прайс-Джонса больного человека

Кривая Прайс-Джонса здорового человека показана на рис. 4.11.

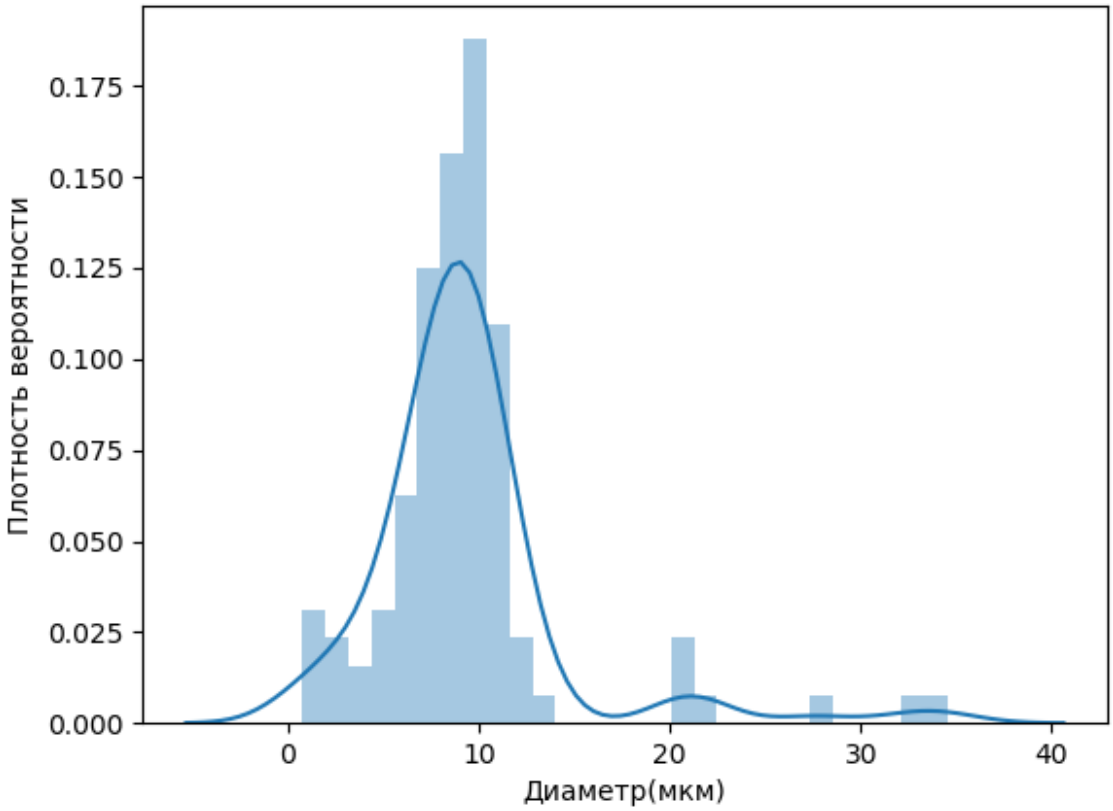

Рис.4.11. Кривая Прайс-Джонса здорового человека

ЗАКЛЮЧЕНИЕ

Кривая Прайса-Джонса у здоровых людей имеет правильную треугольную форму с высокой вершиной и узким основанием. При этом преобладают эритроциты с диаметром 6-8 мкм, которые составляют 70-75% всех эритроцитов. На долю микроцитов (клетки диаметром меньше 6 мкм) и макроцитов (диаметр более 8мкм) приходится приблизительно одинаковое количество 12-15%, ширина кривой отражает степень анизоцитоза, а положение максимума - средний диаметр эритроцита.

При микроцитозе (характерном, например, для железодефицитоной анемии) эритроцитометрическая кривая сдвигается влево, кривая становится ассиметричной, ширина ее увеличивается.

При макроцитозе (например, сопровождающем В12 и фолиеводефицитную анемию) кривая Прайс-Джонса сдвигается вправо, уплощается, основание ее расширяется.

Построение кривой Прайс-Джонса вручную - чрезвычайно трудоемкая процедура. Поэтому разработанная система гематологического анализа позволяем в автоматическом режиме быстро и с высокой точностью построить кривую Прайса-Джонса, что дает возможность увидеть процентное соотношение всех видов красных кровяных телец. Это позволяет достаточно быстро выявить анизоцитоз у больного и подобрать соответствующие методы его лечения.

Разработанный алгоритм обработки медицинских изображенийклеток крови позволил реализовать счет форменных объектов в мазке, по единой их фотографии. Использование предложенных алгоритмов позволило выполнить качественную предварительную обработкумедицинские изображения клеток крови для уменьшения шумов и повышения точности сегментации объектов клеток на классы. Алгоритм пороговой сегментации

изображения и выделения каждой клетки в свой класс показал возможность идентификации и счета форменных элементов клеток крови.

Вычислительные эксперименты по распознаванию форменных объектов на медицинских изображениях клеток крови выполнялись с использованием программного обеспечения, реализованного на языке Python 2.7 и библиотек OpenCV и Seaborn. Результаты вычислительных экспериментов продемонстрировали работоспособность и эффективность разработанных алгоритмов.

Построение кривой Прайс-Джонса вручную – весьма трудоемкая задача. Поэтому разработанная система гематологического анализа позволяет в автоматическом режиме быстро и с высокой точностью построить кривую Прайс-Джонса, что дает возможность увидеть процентное соотношение всех видов красных кровяных телец, что облегчает возможность своевременного выявления аницитоза у больных, подбирает соответствующие методы их лечения, и в целом ускорить выздоровление, сделать жизнь многих пациентов полноценной.

СПИСОКИСПОЛЬЗУЕМОЙЛИТЕРАТУРЫ

1. Contrast Limited Adaptive Histogram Equalization for Qualitative Enhancement of M yocardial Perfusion Images. Engineering [Электронный ресурс], режим доступа - file.scirp.org/pdf/ENG_2013110109155688.pdf, свободный (дата обращения 18.02.2018).
2. Image Thresholding [Электронный ресурс], режим доступа - docs.opencv.org/trunk/d7/d4d/tutorial_py_thresholding.html, свободный (дата обращения 13.04.2018).
3. Iron-deficiency anaemia in children: Its association with gastro-intestinal disease, achlorhydria and haemorrhage. Archives of disease in childhood [Электронныйресурс], режимдоступа - pdfs.semanticscholar.org/6a86/f416daf9c3d90217db7e25cb86273bb1be42.pdf, свободный (датаобращения 10.01.2018).
4. Quantitative Anisocytosis as a Discriminant Between Iron Deficiency and Thalassem. Blood [Электронный ресурс], режим доступа - www.bloodjournal.org/content/bloodjournal/53/2/288.full.pdf?sso-checked=true, свободный (дата обращения 14.05.2018).
5. Red blood cells classification using image processing. Science [Электронныйресурс], режимдоступа - Research studyres.com/doc/17754179/red-blood-cells-classification-using-image, свободный (датаобращения 23.01.2018).
6. The errors of some haematological methods as they are used in a routine laboratory [Электронныйресурс], режимдоступа - jcp.bmj.com/content/jclinpath/1/5/269.full.pdf, свободный (датаобращения 18.05.2018).
7. The variation in the sizes of reb blood cells. British Medical Journal [Электронныйресурс], режимдоступа -

digitalcommons.ohsu.edu/cgi/viewcontent.cgi?article=1062&context=hca-cac, свободный (датаобращения 12.02.2018).

8. Topological Structural Analysis of Digitized Binary Images by Border Following. Computer vision, graphics, and image processing [Электронныйресурс], режимдоступа - download.xuebalib.com/xuebalib.com.17233.pdf, свободный (датаобращения 18.02.2018).
9. Watershed approaches for color image segmentation [Электронныйресурс], режимдоступа - www.gipsa-lab.grenoble-inp.fr/~jocelyn.chanussot/publis/ieee_nsip_99_chanuss_watershed.pdf, свободный (датаобращения 18.02.2018).
10. Beucher, S. Optical Engineering. New York: Marcel Dekker Incorporated [Текст]/ Beucher, S., F. Meyer, - 1994- pp: 433-481.
11. Canny J. A computational approach to edge detection. IEEE Trans. Pattern Anal. Mach. Intell[Текст]/ Canny J. – 1986. – Vol. 8, no. 6. – P. 679-698.
12. Canny J. A computational approach to edge detection,IEEE Trans- ac ons on Pa ern Analys s and Mach ne In ell gence, [Текст]/ Canny J. - 1986, p. 679-698.
13. Gil J. Computing 2-D Min, Median, and Max Filters [Текст]/ J. Gil, Werman // IEEE Trans. Pattern Anal. Mach. Intell. – 1993. – P. 504–507.
14. Gil J. Computing 2-D Min, Median, and Max Filters, IEEE Trans. Pattern Anal. Mach. Intell [Текст]/ Gil J., M. Werman - 1993, Vol 15, Number 5, 504–507.
15. High-performance method for boundary detection in medical images// Academic science – problems and achievements IX: Proceedings of the Conference. North Charleston [Текст]/Soinikova E.S., Ryabihk M.S., Batishchev D.S., Mikhelev V.M., 20-21.06.2016–North Charleston, SC, USA:CreateSpace, 2016, p.93-95.

16. Kirk D. B. Programming Massively Parallel Processors: A Hands-on Approach (Applications of GPU Computing Series). Morgan Kaufmann[Текст]/ Kirk D. B., Hwu W. - 2010.
17. Prentice Hall Series in Artificial Intelligence, Upper Saddle River, NJ: Prentice Hall [Текст]/S. Russell, P. Norvig, J. F. Canny, J. Malik, and D.D. Edwards, Artificial Intelligence: A Modern Approach, 2nd ed., /Pearson Education - 2003.
18. Privacy preserving link analysis on dynamic tweighted graph [Текст]/Y. Duan, J. Wang, M. B. T. Kam, and J. F.Canny, Computational & MathematicalOrganization Theory, vol. 11, no. 2, July 2005, pp. 141-159.
19. Van Herk M. A fast algorithm for local minimum and maximum filters on rectangular and octagonal kernels [Текст]/ Van Herk M. - Pattern Recognition Letters. – 1992. – Vol. 13. – №7. – P. 517–521.
20. Математическая морфология [Электронный ресурс], режим доступа - https://habrahabr.ru/post/113626/, свободный (дата обращения: 12.02.2017).
21. Медицинские изображения [Электронный ресурс] // SCI-ARTICLE. RU. – Электрон. журн. – Режим доступа: http://sci-article.ru/stat.php?i=1413 957877. – Загл. с экрана.
22. Методическое руководство: Общий анализ крови (трактовка результатов исследований, выполненных на гематологических анализаторах) // Ставропольский государственный медицинский университет [Электронный ресурс], режим доступа - stgmu.ru/userfiles/depts/clinical_lab_diagnosis_pe/Obschij_analiz_krovi.rtf, свободный (дата обращения: 12.02.2017).
23. Методы распознавания медицинских изображений для задач компьютерной автоматизированной диагностики [Электронный ресурс] // Современные проблемы науки и образования. – Электрон. журн. – М., [2005–]. – Режим доступа: http://www.science-education.ru/ru/article/view?id=14414. – Загл. с экрана. 3

24. Методы сегментации изображений: автоматическая сегментация [Электронный ресурс] // Компьютерная графика и мультимедиа. – Электрон. журн. – М., – Режим досутпа: http://cgm.computergraphics.ru/content/view/147. – Загл. с экрана.
25. Автоматическая сортировка лейкоцитов мазка крови с использованием методов обучаемых нейронных сетей и watershed [Текст]/Соколинский Б.З., Демьянов В.Л., Медный В.С., Парпара А.А., Пятницкий А.М. // В сб.: Методы микроскопического анализа. М.: Медицинские компьютерные системы, 2009. С. 128–132.
26. Борисовский С.А. Гибридные модели и алгоритмы для анализа сложноструктурированных изображений в интеллектуальных системах медицинского назначения [Текст]/ Борисовский С.А. - Курск, 2012.
27. Виллевальде А.Ю Метод предварительной обработки медицинских малоконтрастных изображений [Текст]/Виллевальде А.Ю., Юлдашев // Информационно-управляющие системы. 2008. № 5(36). С.41– 44.
28. Высокопроизводительный метод обнаружения границ на медицинских изображениях [Текст]/Сойникова Е.С., Рябых М.С., Батищев Д.С., Синюк В.Г., Михелев В.М. // Научный результат. Информационные технологии. 2016. - С. 4-9.
29. Инфраструктура высокопроизводительной компьютерной системы для реализации облачных сервисов хранения и анализа данных персональной медицины [Текст]/Батищев Д.С., Михелев В.М. // Научные ведомости Белгородского государственного университета. Серия: Экономика. Информатика. - Белгород: Изд-во НИУ БелГУ, 2016. - С. 88-92.
30. Использованием методов обучаемых нейронных сетей и watershed [Текст]/Соколинский Б.З., Демьянов В.Л., Медный В.С., Парпара А.А., // В сб.: Методы микроскопического анализа. М.: Медицинские компьютерные системы, 2009. С. 128–132.

31. Липунова Е.А Система красной крови [Текст]/ Липунова Е.А. под ред. Скоркиной М.Ю. - Белгород: Федеральное государственное автономное образовательное учреждение высшего профессионального образования "Белгородский государственный национальный исследовательский университет", 2004.
32. Об одной методике классификации клеток крови и ее программной реализации [Текст]/Беляков В.К., Сухенко Е.П., Захаров А.В., Кольцов П.П., Котович Н.В., Кравченко А.А., Куцаев А.С., Осипов А.С., Кузнецов А.Б. // Программные продукты и системы. – 2014. -№ 4 (108). – С. 46-56.
33. Программное обеспечение интеллектуальной системы классификации форменных элементов крови [Текст]/Томакова Р.А., Филист С.А., Жилин В.В., Борисовский С.А. // Фундаментальные исследования. – 2013. – № 10-2. – С. 303-307.
34. Р. Гонсалес Цифровая обработка изображений[Текст]/ Р. Гонсалес, Р. Вудс, Москва: Техносфера, 2005. – 1072 с.
35. Разработка и испытание автоматизированного комплекса микроскопии [Текст]/Медовый В.С., Пятницкий А.М., Соколинский Б.З., Маркеллов В.В., Федорова Д. С., Федоров И. В. // «Оптический журнал»: Спб. – 2011, с. 66-73.
36. Решение задачи сегментации медицинских изображений с использованием параллельных вычислений [Текст]/Рябых М. С., Сойникова Е. С., Батищев Д.С., Михелёв В.М. // «Приоритетные направления развития науки, техники и технологий: сборник материалов международной научно-практической конференции» (29 февраля 2016 г.) Том II. – Кемерово: ЗапСибНЦ, 2016 – 460 с.
37. Система контроля и управления доступом по биометрическим параметрам [Текст]/М.А. Морозова, О.Б. Салтанова. // Научная сессия ТУСУР–2011: Материалы Всероссийской научно-технической

конференции студентов, аспирантов и молодых ученых, Томск, 4–6 мая 2011 г. – Томск: В-Спектр, 2011: В 6 частях. – Ч. 2. – 348 с.

38. Сойфер, В.А. Методы компьютерной обработки изображений. — 2-е изд. [Текст]/ В.А. Сойфер, испр. — М.:ФИЗМАТЛИТ, 2003 — 784 с.
39. Сравнительное исследование методов анализа изображений [Текст]/Грибков И.В., Захаров А.В., Кольцов П.П., Котович Н.В., Кравченко А.А., Куцаев А.С., Осипов А.С. // - М.: Изд-во НИИСИ РАН, 2005.
40. Яне Б. Цифровая обработка изображений[Текст]/ Яне Б. - Техносфера, 584 с., 2007.

ПРИЛОЖЕНИЕ

scratch.py

```
# -*- coding: utf-8 -*-

importsys
importcv2

importmatplotlib.pyplotasplt
importseaborn

fromLayers import LAYERS

if __name__ == "__main__":
    try:
        imgFileName = sys.argv[1]
    except:
        imgFileName = "dataset/set2/5.jpg"
        #imgFileName = "dataset/set1/1.jpg"

    from Pipeline import Pipeline

    pipeline = Pipeline()
    pipeline.addLayer({"name": "bgr2hlsf",
                       "cls": LAYERS.BGR2HLSF,
                       })
    pipeline.addLayer({"name": "luma",
                       "cls": LAYERS.LumaExtract,
                       "params": {
                           "cindex": 1},
                       })
    # pipeline.addLayer({"name": "blur",
```

```
    pipeline.addLayer({"name": "clahe",
                       "cls": LAYERS.CLAHE,
                       "params": {"clipLimit": 3.0,
                                  "tileGridSize": (7, 7)},
                       })
    pipeline.addLayer({"name": "thresh",
                       "cls": LAYERS.Threshold
                       })
    pipeline.addLayer({"name": "disttr",
                       "cls": LAYERS.DistanceTransform
                       })
    pipeline.addLayer({"name": "norm",
                       "cls": LAYERS.Normalize
})
pipeline.addLayer({"name": "thresh_norm",
                       "cls": LAYERS.Threshold,
                       "params": {
                          "maxval": 1.,
                          "type": cv2.THRESH_BINARY}
                       })
    pipeline.addLayer({"name": "dilate",
                       "cls": LAYERS.Dilate
                       })
    pipeline.addLayer({"name": "fcontours",
                       "cls": LAYERS.FindContours
                       })
    pipeline.addLayer({"name": "dcontours",
                       "cls": LAYERS.DrawContours
                       })
    pipeline.addLayer({"name": "watersh",
                       "cls": LAYERS.Watershed,
                       "input": {"srcimage": "pipe",
                                 "image": "thresh",
```

```
                    "markers": "dcontours"}
                })

    pipeline.addLayer({"name": "csquares",
                "cls": LAYERS.CalcContours,
                "input": "fcontours",
                "params": {
                   "lowbound": 100,
                   "upperbound": 8000}
                })

    pipeline.input = imgFileName
    pipeline.run()

    squares = pipeline.layers["csquares"].output
    diams = [0.154 * 0.154 * x / 3.14 for x in squares]
    seaborn.distplot(diams, hist=True, kde=True)
plt.ylabel(u"Плотность вероятности")
    plt.xlabel(u"Диаметр(мкм)")
plt.show()

    cv2.imshow("out", pipeline.layers["watersh"].output)
    cv2.waitKey(0)
```

BGR2HLSF.py

```
# -*- coding: utf-8 -*-

import cv2

from BaseLayer import BaseLayer

class BGR2HLSF(BaseLayer):
    """
```

```
    def process(self):
        self._output = cv2.cvtColor(self.input, cv2.COLOR_BGR2HLS_FULL)
        pass
```

BaseLayer.py

```
class BaseLayer(object):
    """
    base init
    def __init__(self, **kwargs):
        self._input, self._params = None, {}
        if "input" in kwargs:
            self._input = kwargs.get("input")
        if "params" in kwargs:
            self._params = kwargs.get("params")

        self._output = None

        self.validateInput()
        self.configure()
        self.prepare()

    @property
    def input(self):
        return self._input

    @input.setter
    def input(self, value):
        self._input = value
        self.validateInput()
        self.prepare()

    @property
    def params(self):
```

```
    return self._params

@params.setter
def params(self, value):
    self._params = value
    self.configure()
    self.prepare()

@property
def output(self):
    return self._output

@output.setter
def output(self, value):
    raise ValueError("output is readonly property")

def validateInput(self):
    pass

def param(self, name, default):
    v = self._params.pop(name, default)
    self._params[name] = v

def configure(self):
    pass

def prepare(self):
    pass

def process(self):
    pass
```

Blur.py

```
import cv2

from LumaLayer import LumaLayer

class Blur(LumaLayer):

    def configure(self):
        self.param("ksize", (5, 5))
        self.param("sigmaX", 0)

    def process(self):
        # like noise filter
        self._output = cv2.GaussianBlur(self._input,
                                        self.params["ksize"],
                                        self.params["sigmaX"])
```

CLAHE.py

```
import cv2

from LumaLayer import LumaLayer

class CLAHE(LumaLayer):
    """
    ss
    """
    def __init__(self, **kwargs):
        self.CLAHE = None
        super(CLAHE, self).__init__(**kwargs)

    def configure(self):
```

```
        self.param("clipLimit", 3.0)
        self.param("tileGridSize", (7, 7))

    def prepare(self):
        self.CLAHE = cv2.createCLAHE(clipLimit=self.params["clipLimit"],
                                     tileGridSize=self.params["tileGridSize"])

    def process(self):
        self._output = self.CLAHE.apply(self._input)
```

CalcContours.py

```
import numpy as np
import cv2

from BaseLayer import BaseLayer

class CalcContours(BaseLayer):

    def validateInput(self):
        assert "contours" in self._input
        assert "hierarchy" in self._input

    def configure(self):
        self.param("lowbound", 100)
        self.param("upperbound", 8000)

    def prepare(self):
        self.contours = self._input["contours"]
        self.hierarchy = self._input["hierarchy"]
        self.squares = []

    def process(self):
        for contour in self.contours:
```

```
                area = cv2.contourArea(contour)
                if area > 100 and area < 8000:
                    self.squares.append(area)

            self._output = self.squares
```

CleanBackground.py

```
import cv2

from LumaLayer import LumaLayer

class CleanBackground(LumaLayer):
    def configure(self):
        self.param("meanPixelsPercent", 0.9)
        self.param("bgColor", 255)

    def process(self):
        # bg remove
        # like blanking pixels with luminance more than n of mean value
        self._input[self._input < self._input.mean() * self.params["meanPixelsPercent"]] =
self.params["bgColor"]

        self._output = self._input
```

Dilate.py

```
import numpy as np
import cv2

from LumaLayer import LumaLayer

class Dilate(LumaLayer):
```

```
    def configure(self):
        self.param("kernel", np.ones((3, 3), np.uint8))

def process(self):
self._output = cv2.dilate(self._input,
                          self.params["kernel"])
```

DistanceTransform.py

```
import numpy as np
import cv2

from LumaLayer import LumaLayer

class DistanceTransform(LumaLayer):

    def configure(self):
        self.param("type", cv2.DIST_L2)
        self.param("maskSize", 3)

    def process(self):
        self._output = cv2.distanceTransform(self._input,
                                             self.params["type"],
                                             self.params["maskSize"])
```

DrawContours.py

```
import numpy as np
import cv2

from BaseLayer import BaseLayer

class DrawContours(BaseLayer):
```

```
    def validateInput(self):
        assert "image" in self._input
        assert "contours" in self._input
        #assert "hierarchy" in self._input

    def prepare(self):
        self.image = self._input["image"]
        self.contours = self._input["contours"]
        #self.hierarchy = self._input["hierarchy"]
        self.markers = np.zeros(self.image.shape, dtype=np.int32)

    def process(self):

        for idx in xrange(1, len(self.contours)):
            cv2.drawContours(self.markers, self.contours, idx - 1, idx)

        self._output = self.markers
```

FindContours.py

```
import numpy as np
    import cv2

    from LumaLayer import LumaLayer

    class FindContours(LumaLayer):
        # TODO: set (0,0) pixel to background color, thus Suzuki85 algo should work fine and find left-top-most object

        def configure(self):
            self.param("mode", cv2.RETR_EXTERNAL)  # cv2.RETR_EXTERNAL
            self.param("method", cv2.CHAIN_APPROX_SIMPLE)

        def prepare(self):
```

```
            self._input = np.uint8(self._input.copy())


        def process(self):
        image, contours, hierarchy = cv2.findContours(self._input,
        self.params["mode"],
        self.params["method"])


            self._output = {"image": image,
                        "contours": contours,
                        "hierarchy": hierarchy}
```

LumaExtract.py

```
import cv2

        from BaseLayer import BaseLayer

        class LumaExtract(BaseLayer):
           """"""

           def configure(self):
              self.param("cindex", 0)

           def process(self):
              self._output = cv2.split(self.input)[self.params["cindex"]]
```

Normalize.py

```
import numpy as np
        import cv2

        from LumaLayer import LumaLayer

        class Normalize(LumaLayer):
```

```
    def configure(self):
        self.param("alpha", 0.)
        self.param("beta", 1.)
        self.param("type", cv2.NORM_MINMAX)

    def process(self):
    self._output = cv2.normalize(self._input,
                                 self._input,
                                 alpha=self.params["alpha"],
    beta=self.params["beta"],
    norm_type=self.params["type"])
```

Watershed.py

```
import numpy as np
    import cv2

    from LumaLayer import LumaLayer

    class Watershed(LumaLayer):

        def validateInput(self):
            assert "srcimage" in self._input
            assert "image" in self._input
            assert "markers" in self._input

        def configure(self):
            self.param("basecolor", (255, 255, 255))

        def prepare(self):
            self.srcimage = self.input["srcimage"]
            self.image = self.input["image"]
            self.markers = self.input["markers"]
```

```
        self.colors = np.int32(np.random.randint(100, 254, size=(len(self.markers), 3)))


    def process(self):
        cv2.watershed(self.srcimage, self.markers)


        for i in xrange(0, self.markers.max()):
            self.srcimage[self.markers == i] = self.colors[i]


self._output = self.srcimage
```

Выпускная квалификационная работа выполнена мной совершенно самостоятельно. Все использованные в работе материалы и концепции из опубликованной научной литературы и других источников имеют ссылки на них.

«___» _____ _____ г.

_____ _____
(подпись) (Ф.И.О.)