Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования

**«КУБАНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»**

**(ФГБОУ ВО «КубГУ»)**

**Факультет компьютерных технологий и прикладной математики**

**Кафедра вычислительных технологий**

**ИНДИВИДУАЛЬНАЯ РАБОТА №1**

**Дисциплина: Алгоритмы цифровой обработки мультимедиа**

Работу выполнили: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ А. А. Сапунова

А. Н. Мирошниченко

Направление подготовки: 02.03.02 Фундаментальная информатика и информационные технологии

Преподаватель: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ А. С. Жук

Цель работы: анализ применимости методов выявления границ объектов на изображении заданного типа.

Задание 1. Выбрать тип изображений для анализа

Для анализа были выбраны снимки мазков крови человека. Для данного типа изображений поиск контуров может быть применен как часть задачи для возможного диагноза путем количественного описания характеристик клеток и ядер крови.

Задание 2. подготовить набор данных на проведения анализа

Набор данных для анализа представлены на рис. 1–4.

Изображение выглядит как ткань

Автоматически созданное описание

Рисунок 1 – Снимок крови человека №1

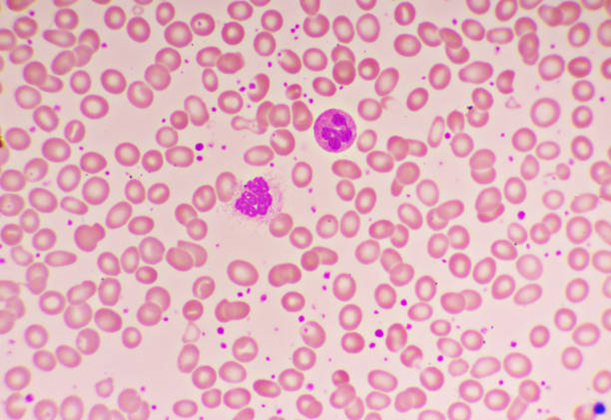


Рисунок 2 – Снимок крови человека №2

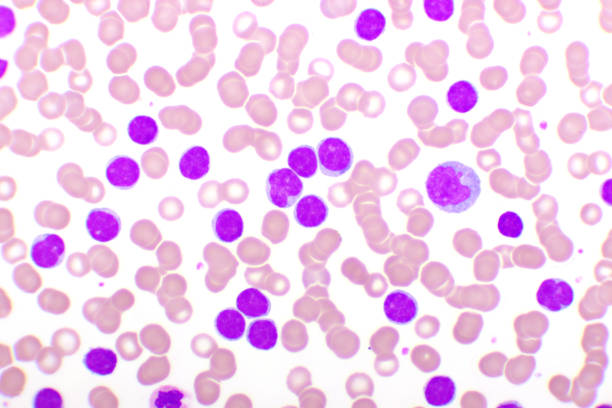


Рисунок 3 – Снимок крови человека №3

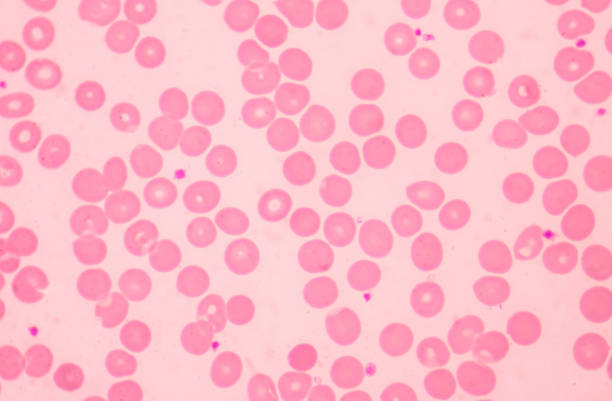


Рисунок 4 – Снимок крови человека №4

Задание 3. протестировать алгоритм Канни и выявить оптимальные параметры (размытие, пороги фильтрации) для выявления границ;

Оптимальными параметрами для пороговой фильтрации для данного типа изображения является (Рис. 5-6):

low\_level = max\_grad // 6

high\_level = max\_grad // 3

, где low\_level – нижний порог фильтрации,

high\_level – верхний порог фильтрации,

max\_grad – максимальное значение градиента

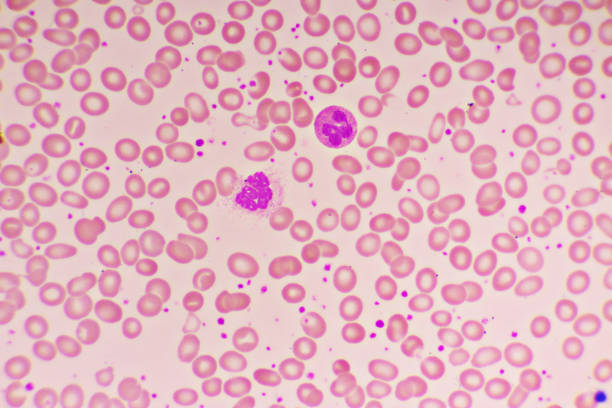


Рисунок 5 – Исходное изображение

Изображение выглядит как природа, белый, делает, рука

Автоматически созданное описание

Рисунок 6 – Контуры при low\_level = max\_grad //6, high\_level = max\_grad // 3

При больших значениях делителя изображение получается слишком детализированным (Рис. 7-8).

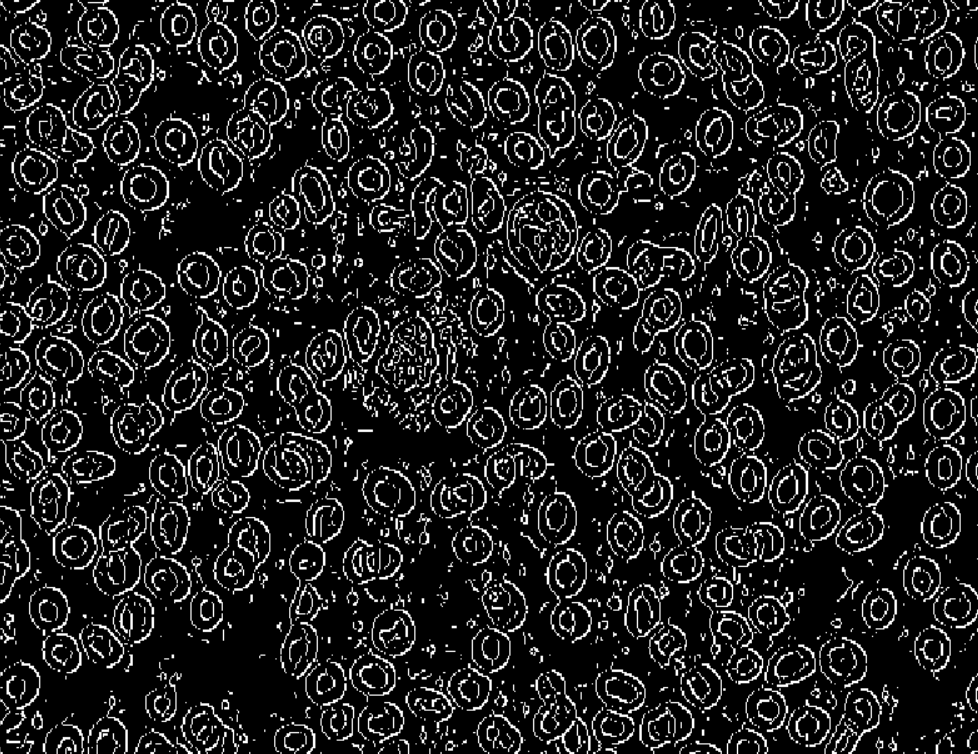


Рисунок 7 – Контуры при low\_level = max\_grad //70, high\_level = max\_grad // 60

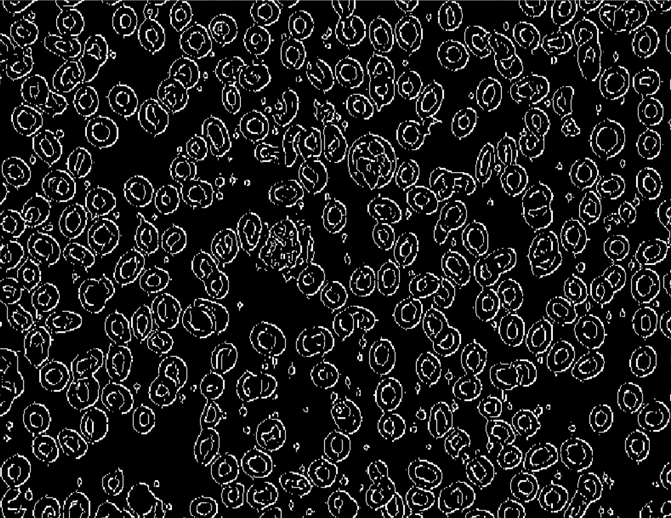


Рисунок 8 – Контуры при low\_level = max\_grad //20, high\_level = max\_grad // 10

Оптимальными параметрами для матрицы сверки является N = 3 (Рис. 9 -10).

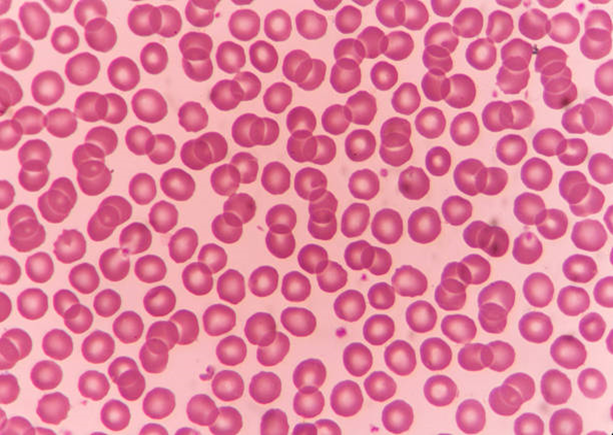


Рисунок 9 – Исходное изображение

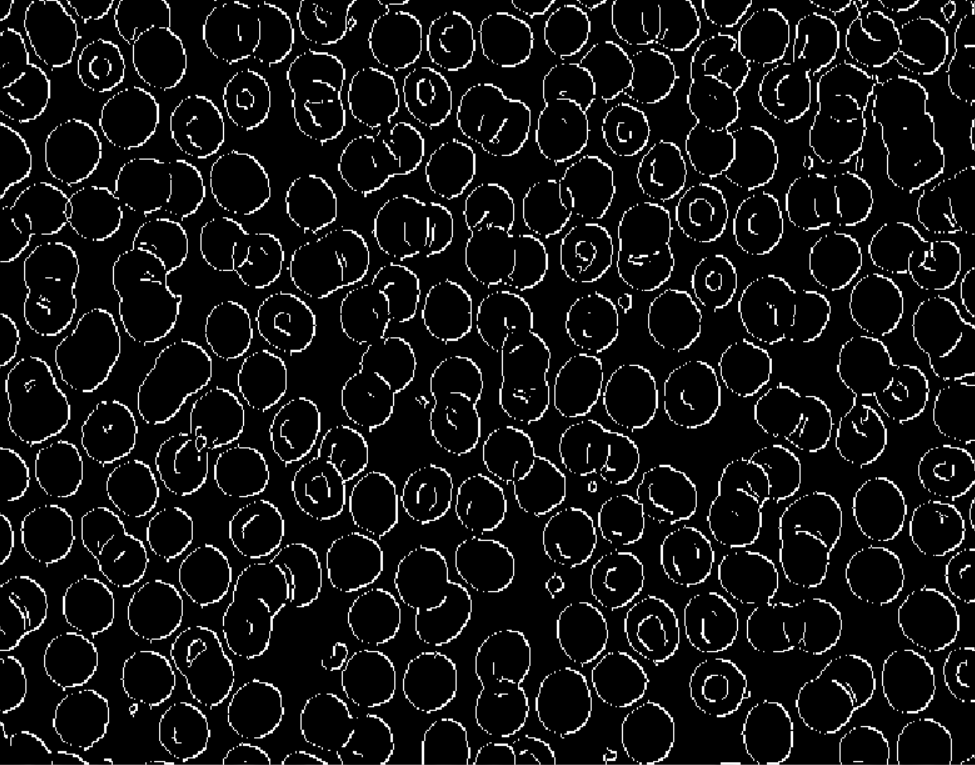


Рисунок 10 – Контуры при N=3

При больших значениях N изображение получается слишком детализированным (Рис. 11-12).

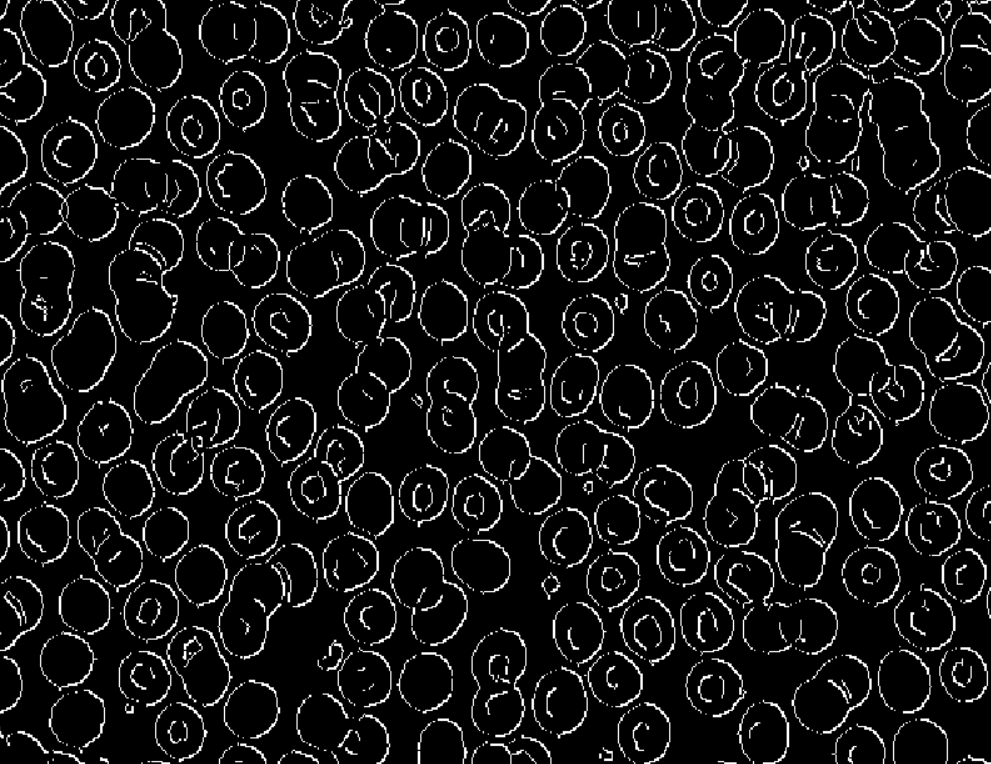


Рисунок 11 – Контуры при N=7

Изображение выглядит как стена, белый

Автоматически созданное описание

Рисунок 12 – Контуры при N=13

Оптимальные параметры для σ является 0.5 (Рис.13).

Изображение выглядит как делает, трюк, белый, воздух

Автоматически созданное описание

Рисунок 13 – Контуры при σ=0.5

При σ большим, чем 0.5 изображения становится слишком детализированным (Рис.14 -15).

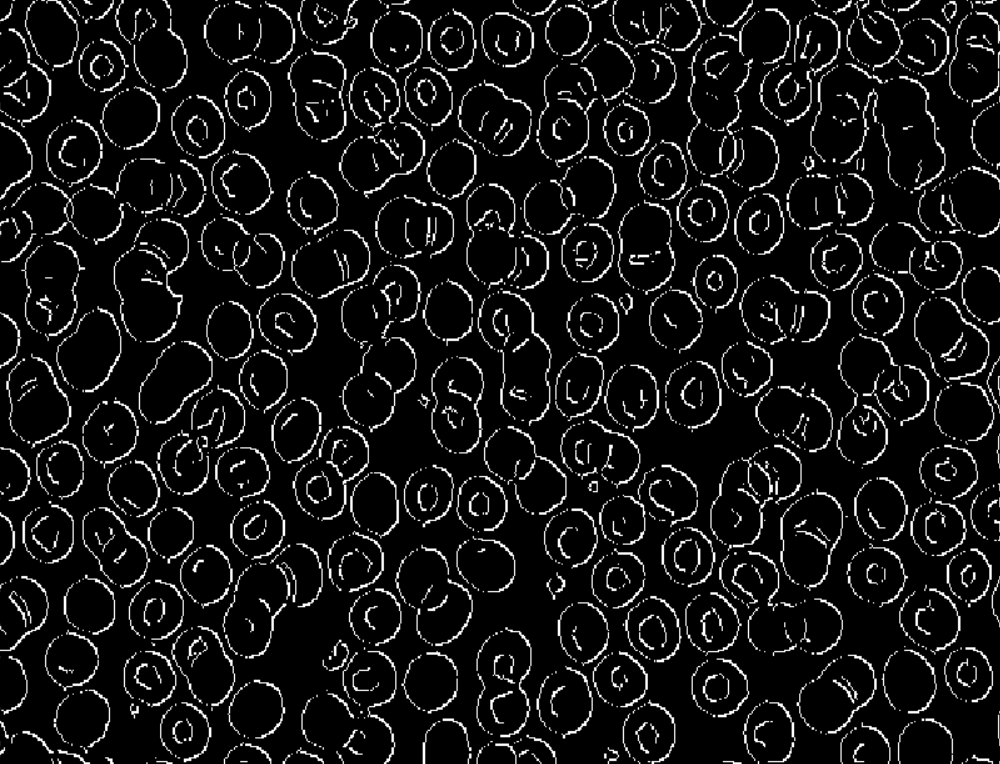


Рисунок 14 – Контуры при σ=1.5

Изображение выглядит как стена, белый

Автоматически созданное описание

Рисунок 15 – Контуры при σ=1

Задание 4. Протестировать алгоритм Канни, заменив оператор Собеля на два других оператора, для каждого из операторов подобрать оптимальные параметры алгоритма.

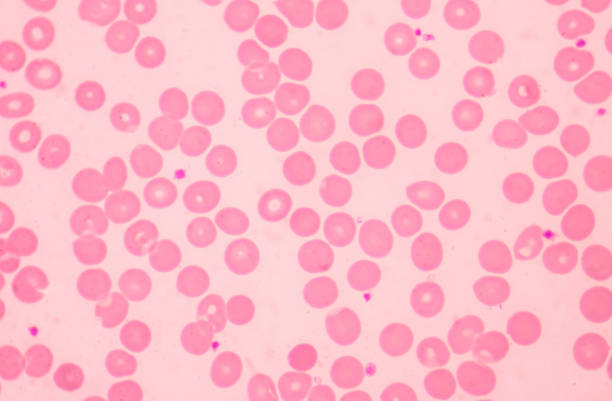


Рисунок 16 – Исходное изображение

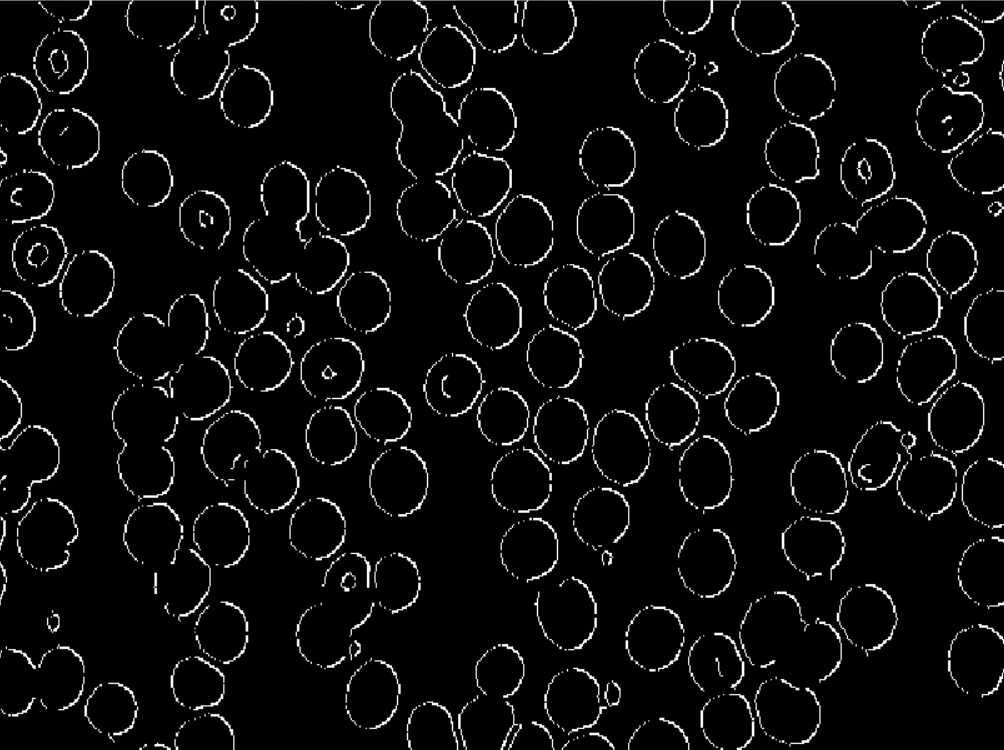


Рисунок 17 – Оператор Собеля

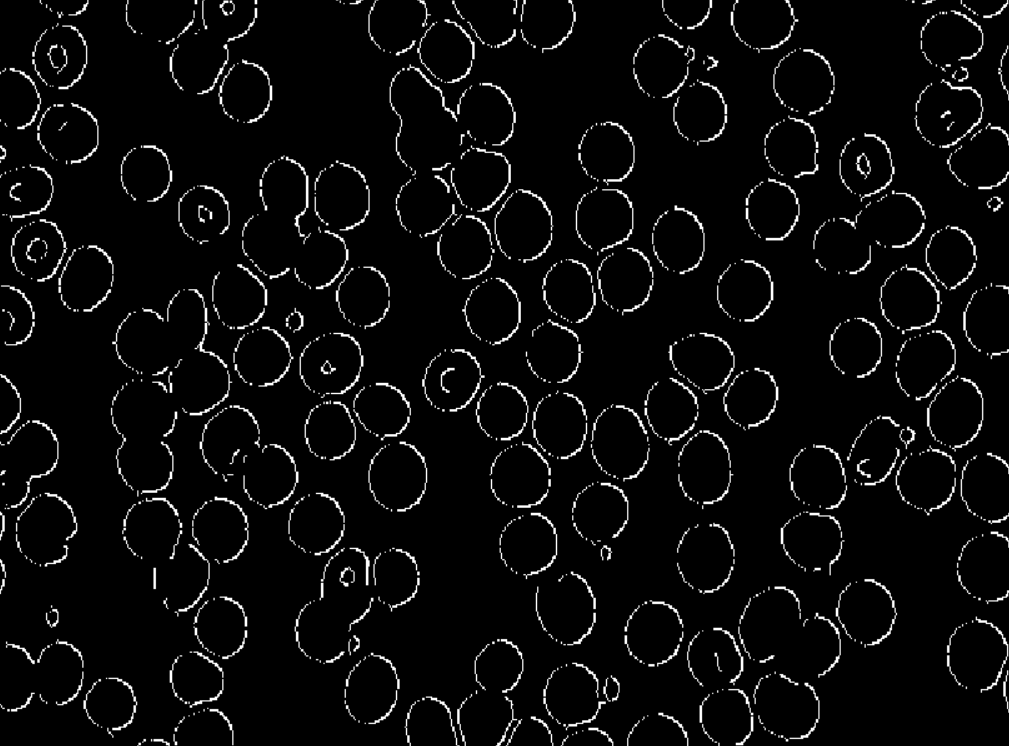


Рисунок 18 – Оператор Прюитта

Изображение выглядит как природа, наружный объект, паутина

Автоматически созданное описание

Рисунок 19 – Оператор Робертса

Задание 5. Попробовать реализовать альтернативный способ выявления границ для изображений данного типа, возможно воспользоваться готовыми библиотеками, лучше реализовать алгоритм самостоятельно.

В качестве альтернативного способа выявления границ был выбран алгоритм водоразделов.

Алгоритм:

1. В местах локального минимума образуем «отверстия», через которые вода начнет заполнять трехмерную поверхность.

2. Если вода с двух сторон гребня готова объединиться в один бассейн, устанавливаем перегородку.

3. Когда над водой останутся только загородки, останавливаем алгоритм.

Задание 6. Провести сравнительный анализ примененного алгоритма с алгоритмом Канни с оптимальными параметрами, учесть не только качество выявления, но и скорость работы алгоритмов.

Сравнительный анализ приведен на рис. 17–18.



Рисунок 20 - Метод водоразделов (Время работы: 0.016590)

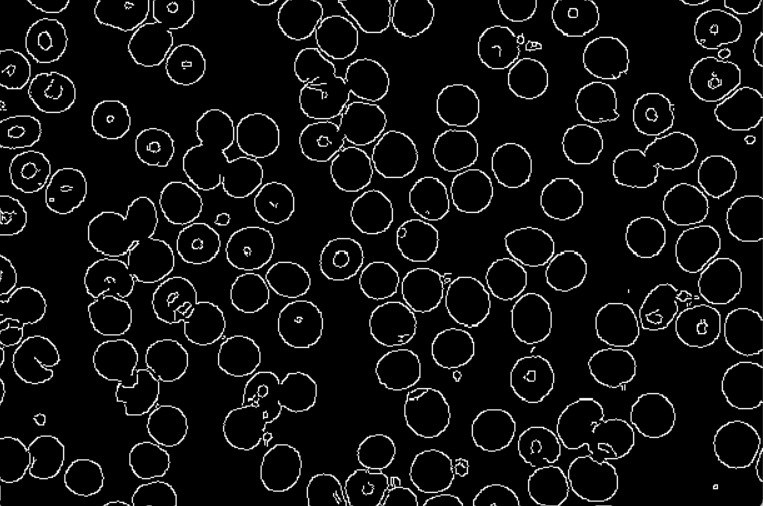


Рисунок 21 – Алгоритм Канни (Время работы: 0.00402

**Листинг**

import cv2 # task1  
import numpy as np  
import requests  
import imutils  
  
  
# task 2  
#  
  
# IMREAD\_GRAYSCALE  
# IMREAD\_ANYDEPTH  
# cv2.IMREAD\_REDUCED\_COLOR\_4  
def read\_photo():  
 img = cv2.imread(r'.\1604650136\_1.jpg')  
 # WINDOW\_NORMAL  
 # WINDOW\_AUTOSIZE  
 # WINDOW\_FREERATIO  
 cv2.namedWindow('Display window')  
 cv2.imshow('Display window', img)  
 cv2.waitKey(0)  
 cv2.destroyAllWindows()  
  
  
# task 3  
# COLOR\_RGB2HSV COLOR\_BGR2HSV  
def read\_video():  
 cap = cv2.VideoCapture(r'.\1.mp4', cv2.CAP\_ANY)  
 cv2.namedWindow('frame', cv2.WINDOW\_NORMAL)  
 cv2.resizeWindow("frame", 500, 500)  
  
 while True:  
 ret, frame = cap.read()  
 if not ret:  
 break  
  
 # cv2.imshow('frame', frame)  
  
 # Черно-белое  
 gray = cv2.cvtColor(frame, cv2.COLOR\_BGR2GRAY)  
 cv2.imshow('frame', gray)  
  
 if cv2.waitKey(10) & 0xFF == 27:  
 break  
  
 cap.release()  
 cv2.destroyAllWindows()  
  
  
# task 4  
def read\_video\_write\_to\_file():  
 video = cv2.VideoCapture(r'.\1.mp4', cv2.CAP\_ANY)  
  
 w = int(video.get(cv2.CAP\_PROP\_FRAME\_WIDTH))  
 h = int(video.get(cv2.CAP\_PROP\_FRAME\_HEIGHT))  
 fourcc = cv2.VideoWriter\_fourcc(\*'XVID')  
 video\_writer = cv2.VideoWriter("output.mov", fourcc, 25, (w, h))  
  
 while True:  
 ok, img = video.read()  
 if not ok:  
 break  
  
 video\_writer.write(img)  
 if cv2.waitKey(1) & 0xFF == ord('q'):  
 break  
  
 video.release()  
 cv2.destroyAllWindows()  
  
  
# task 5  
def read\_ip\_write\_to\_file():  
 video = cv2.VideoCapture(0)  
 w = int(video.get(cv2.CAP\_PROP\_FRAME\_WIDTH))  
 h = int(video.get(cv2.CAP\_PROP\_FRAME\_HEIGHT))  
 fourcc = cv2.VideoWriter\_fourcc(\*'XVID')  
 video\_writer = cv2.VideoWriter("output.mov", fourcc, 25, (w, h))  
  
 while True:  
 ok, img = video.read()  
 cv2.imshow('img', img)  
 video\_writer.write(img)  
 if cv2.waitKey(1) & 0xFF == ord('q'):  
 break  
  
 video.release()  
 cv2.destroyAllWindows()  
  
  
def print\_cam():  
 cap = cv2.VideoCapture(0)  
 cap.set(3, 640)  
 cap.set(4, 480)  
 while True:  
 ret, frame = cap.read()  
 gray = cv2.cvtColor(frame, cv2.COLOR\_BGR2GRAY)  
  
 cv2.imshow('frame', gray)  
 if cv2.waitKey(1) & 0xFF == 27:  
 break  
  
 cap.release()  
 cv2.destroyAllWindows()  
  
  
# task 6  
def phone\_camera():  
 video = cv2.VideoCapture(1)  
  
 w = int(video.get(cv2.CAP\_PROP\_FRAME\_WIDTH))  
 h = int(video.get(cv2.CAP\_PROP\_FRAME\_HEIGHT))  
 fourcc = cv2.VideoWriter\_fourcc(\*'XVID')  
 video\_writer = cv2.VideoWriter("output2.mov", fourcc, 25, (w, h))  
  
 while True:  
 rez, img = video.read()  
 cv2.imshow('Camera', img)  
 video\_writer.write(img)  
  
 if cv2.waitKey(1) & 0xFF == ord('q'):  
 break  
  
 # cv2.destroyAllWindows()  
 # cv2.waitKey(0)  
  
 video.release()  
 cv2.destroyAllWindows()