**Липецкий государственный технический университет**

Факультет автоматизации и информатики

Кафедра автоматизированных систем управления

Организация графических систем  
Лабораторная работа №2

«Реализация построения контуров изображения на основе детектора Canny»

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Студент | | | | |  | |  |  | | Янин С.А. | | |  |
| Группа |  | М-АС-19 | | | |  | | |  | |  |
|  | | |  |  | | | |  |  | | | |  |
| Руководитель | | |  |  | | | |  | Кургасов В.В. | | | |  |

Липецк 2020 г.

**Задание**

На любом удобном студенту языке программирования разработать программу, осуществляющую применение детектора границ Канни к изображениям с последующим сохранением. Должна быть предусмотрена возможность преобразования файла векторного формата в растровое изображение для дальнейшей обработки. Вычисления при осуществлении функционала должны быть возложены на графический процессор.

# Оглавление

Оглавление 3

Введение 4

1 Теоретические сведения 6

1.1 Детектор границ Канни 6

1.2 Язык реализации 9

2 Исходный код 11

3 Результаты работы программы 12

Вывод 13

# Введение

Анализ изображения — это процесс выделения нужной информации из изображения с помощью автоматических систем. Системы анализа не ограничиваются разделением областей сцены на фиксированное число классов. Они предназначены для описания сложных сцен, разнообразие которых может быть настолько большим, что их нельзя описать с помощью заранее заданных терминов. В системе анализа также могут использоваться методы искусственного интеллекта для управления различными блоками системы и организации эффективного доступа к базе априорных сведений об объектах. Признак изображения — это его простейшая характеристика или свойство. Некоторые признаки являются естественными в том смысле, что они устанавливаются визуальным анализом изображения, тогда как другие, так называемые искусственные признаки, получаются в результате его специальной обработки и измерений. Естественные признаки: светлота (яркость), текстура различных областей изображения и форма контуров объектов. Обычно анализ изображения включает в себя получение внешнего контура изображенных объектов и запись координат точек этого контура. Чаще всего требуется получить внешний контур в виде замкнутой кривой или совокупности отрезков дуг.

Существует целый ряд алгоритмов, решающих задачу фильтрации контуров, такие как операторы Собеля, Лапласа, Прюитта, Робертса и Канни. Оператор Канни использует многоступенчатый алгоритм для обнаружения широкого спектра границ в изображениях [3, 4]. Он не ограничивается вычислением градиента сглаженного изображения. В контуре границы оставляются только точки максимума градиента изображения, а не максимальные точки, лежащие рядом с границей, удаляются. Здесь также используется информация о направлении границы для того, чтобы удалять точки именно рядом с границей и не разрывать саму границу вблизи локальных максимумов градиента. Затем с помощью двух порогов удаляются слабые границы. Фрагмент границы при этом обрабатывается как целое. Если значение градиента где-нибудь на прослеживаемом фрагменте превысит верхний порог, то этот фрагмент остается также «допустимой» границей и в тех местах, где значение градиента падает ниже этого порога, до тех пор, пока она не станет ниже нижнего порога. Если же на всем фрагменте нет ни одной точки со значением, большим верхнего порога, то он удаляется.

В данной лабораторной работе мы реализуем детектор границ Канни. Языком реализации в данной работе выступает JavaScript.

# 1 Теоретические сведения

# 1.1 Детектор границ Канни

Контурный анализ — это область науки, посвященная обработке изображения, содержащая в себе набор алгоритмов и методов по нахождению границ (контуров) объектов и работе с границами объектов на изображении.

Существует множество различных алгоритмов поисков контуров, таких как:

* Оператор Робертса;
* Оператор Прюита;
* Оператор Собеля;
* Детектор границ Канни.

При работе с методами контурного анализа нельзя наверняка сказать, какой из них будет работать лучше на определенной выборке изображений, однако в силу наличия детектора границ Канни в библиотеке OpenCV, а также на основании его популярности в научном мире, будет рассмотрен именно этот алгоритм поиска контуров на изображении.

Детектор границ Канни — оператор для поиска границ объектов на изображении. Был изобретен Джоном Канни в 1986 и использует многоступенчатый алгоритм для поиска широкого спектра границ на изображениях. Работу алгоритма можно представить в виде последовательности шагов:

1. Сглаживание;
2. Поиск градиентов;
3. Подавление “немаксимумов”;
4. Двойная пороговая фильтрация;
5. Трассировка области неоднозначности.

Рассмотрим каждый из пунктов подробнее.

1. Сглаживание

Кадры, которые подаются на вход детектору в общем случае имеют множество шумов. Шумы мешают детектору найти истинные границы объектов, поэтому изначально следует сгладить изображение. В детекторе границ Канни используется фильтр Гаусса с ядром N × N. Фильтр Гаусса представляет из себя матрицу свертки заполненную по закону нормального распределения.

2. Поиск градиентов

Метод основан на перепадах яркостей изображения, для поиска векторов направлений (градиентов) смены яркостей используется оператор Собеля. Изображение сворачивается в вертикальном и горизонтальном направлениях при помощи двух сепарабельных фильтров ядра 3 × 3. Суть этих свёрток заключается в поиске приближенных значений производных в соответствующих направлениях. Имеется исходное изображение A , Gx и Gy — результирующие изображения, где каждая точка содержит приближенные производные по горизонтальному и вертикальному направлениям соответственно. Они вычисляются следующим образом:

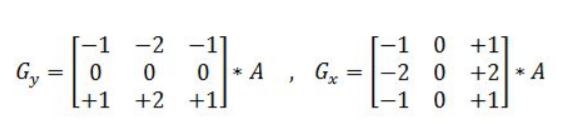


Рисунок 1 – Gx и Gy — результирующие изображения

При этом приближенное значение градиента можно вычислить по формуле:

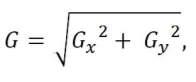


Рисунок 2 – Приближенное значение градиента

а направление градиента по формуле:

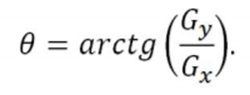


Рисунок 3 – Направление градиента

Потенциальными границами назначаются те точки изображения, в которых градиент принимает наибольшее значение.

3. Подавление “немаксимумов”

После того как определены все значения градиентов и их направления, находятся те пиксели, в которых достигается локальный максимум. Хорошей иллюстрацией этого процесса является рисунк 4.

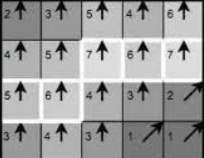


Рисунок 4 – Иллюстрация выбора локального максимума

Области, обведенные белым цветом, применяются как потенциальные границы.

4. Двойная пороговая фильтрация

Оператор использует так называемые пороги для определения существования границы в данном пикселе изображения. Их значения лежат в промежутке от 0 до 255. Чем меньше задается порог, тем больше будет находится границ, но в то же время результат станет более восприимчив к шуму, выделяя лишнее. Если же задается высокий порог, то это может привести к тому, что будут проигнорированы слабые края. Алгоритм выделения границ Канни использует два порога фильтрации: если значение пикселя выше верхней границы — он принимает максимальное значение (граница считается достоверной), если ниже — пиксель подавляется, точки со значением, попадающим в диапазон между порогов, принимают фиксированное среднее значение (они будут уточнены на следующем этапе).

5. Трассировка области неоднозначности

На данном этапе анализируются группы пикселей, попавших в среднюю область. Точки, лежащие по соседству с границей, присоединяются к ней, остальные — отбрасываются.

Данная работа реализована на языке JS и фреймворке Vue.js для быстрого и удобного написания пользовательского интерфейса. Сам алгоритм Канни реализован на шейдерном языке GLSL так как он проще в использовании по сравнению с другими средствами использования вычислительных средств видеокарты (например, CUDA).

# 1.2 Язык реализации

JavaScript — динамический, интерпретируемый язык со слабой типизацией, обычно используемый для написания скриптов на стороне клиента. JavaScript поддерживается всеми современными браузерами и позволяет выполнять на стороне клиента достаточно сложные вычисления. Для JS существует большое количество фреймворков для простого рендеринга 3D-сцен, отрисовки пользовательских интерфейсов. Один из таких фреймворков – Vue.js. Vue.js — JavaScript-фреймворк с открытым исходным кодом для создания пользовательских интерфейсов. Легко интегрируется в проекты с использованием других JavaScript-библиотек. Может функционировать как веб-фреймворк для разработки одностраничных приложений в реактивном стиле. В отличие от фреймворков-монолитов, Vue создан пригодным для постепенного внедрения. Его ядро в первую очередь решает задачи уровня представления (view), что упрощает интеграцию с другими библиотеками и существующими проектами.

Кроме того, современные браузеры также поддерживают и использование GPU API WebGL. WebGL предполагает использование языка шейдеров GLSL, который имеет много общего с С. GLSL (OpenGL Shading Language, Graphics Library Shader Language) — язык высокого уровня для программирования шейдеров. Разработан для выполнения математики, которая обычно требуется для выполнения растеризации графики. Синтаксис языка базируется на языке программирования ANSI C, однако, из-за его специфической направленности, из него были исключены многие возможности, для упрощения языка и повышения производительности. В язык включены дополнительные функции и типы данных, например для работы с векторами и матрицами. Основное преимущество GLSL перед другими шейдерными языками — переносимость кода между платформами и ОС. Язык GLSL используется в OpenGL, в OpenGL ES и WebGL используется язык GLSL ES (OpenGL ES Shading Language).

# 2 Исходный код

Файл «index.html»

<!doctype html>  
<html lang**="en"**>  
<head>  
 <meta charset**="UTF-8"**>  
 <meta name**="viewport"** content**="width=device-width, user-scalable=no, initial-scale=1.0, maximum-scale=1.0, minimum-scale=1.0"**>  
 <meta http-equiv**="X-UA-Compatible"** content**="ie=edge"**>  
 <title>**Лабораторная №2**</title>  
 <link href**="css/app.css"** rel**="stylesheet"**>  
</head>  
<body>  
 <div class**="image"**>  
 <input id**="image-file"** class**="image-input"** type**="file"** accept**="image/\*"**>  
 <label for**="image-file"**>**Выберите файл**</label>  
 <img id**="src-image"** style**="***margin-top*: 20**px"** src**=""** alt**=""**>  
 <canvas id**="the-canvas"** width**="300"**/>  
 </div>  
 <script src**="js/opencv.js"**></script>  
 <script src**="js/app.js"**></script>  
</body>  
</html>

Файл «app.js»

(function() {  
 var fileUploadEl = *document*.getElementById('image-file'),  
 srcImgEl = *document*.getElementById('src-image')  
  
 fileUploadEl.addEventListener("change", function (*e*) {  
 srcImgEl.src = *URL*.createObjectURL(*e*.target.files[0]);  
 }, false);  
  
 srcImgEl.onload = function () {  
 var src = cv.imread(srcImgEl); // load the image from <img>  
 var dst = new cv.Mat();  
  
 cv.cvtColor(src, src, cv.COLOR\_RGB2GRAY, 0);  
  
 cv.Canny(src, dst, 50, 100, 3, false); // You can try more different parameters  
 cv.imshow('the-canvas', dst); // display the output to canvas  
  
 src.delete(); // remember to free the memory  
 dst.delete();  
 }  
  
})()

# 3 Результаты работы программы

Для запуска приложения необходимо открыть файл index.html в любом из удобных браузеров, нажать на кнопку «Выберите файл» и в открывшемся диалоговом окне выбрать фотографию для дальнейшей обработки.

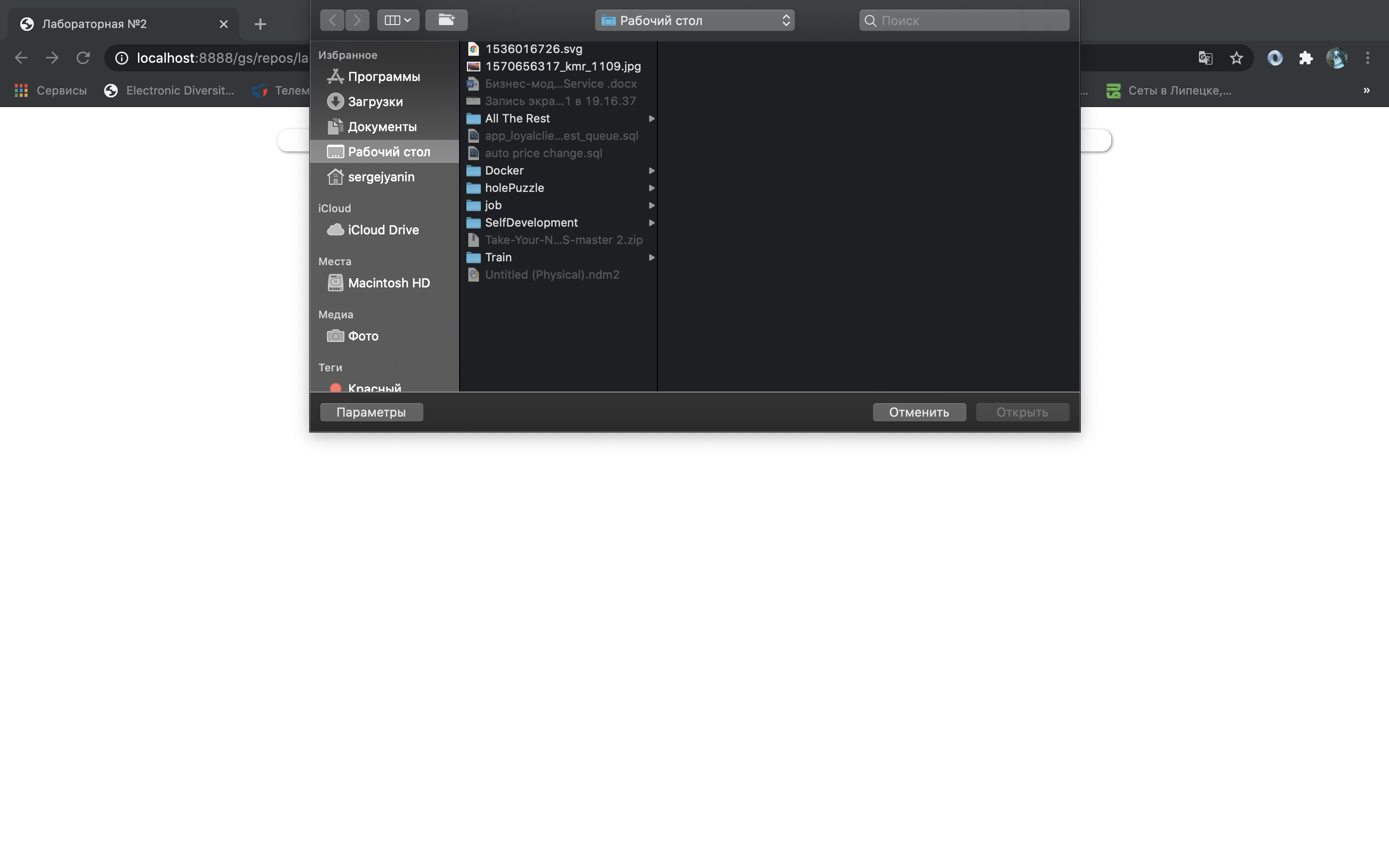


Рисунок 5 – Загрузка исходного изображения

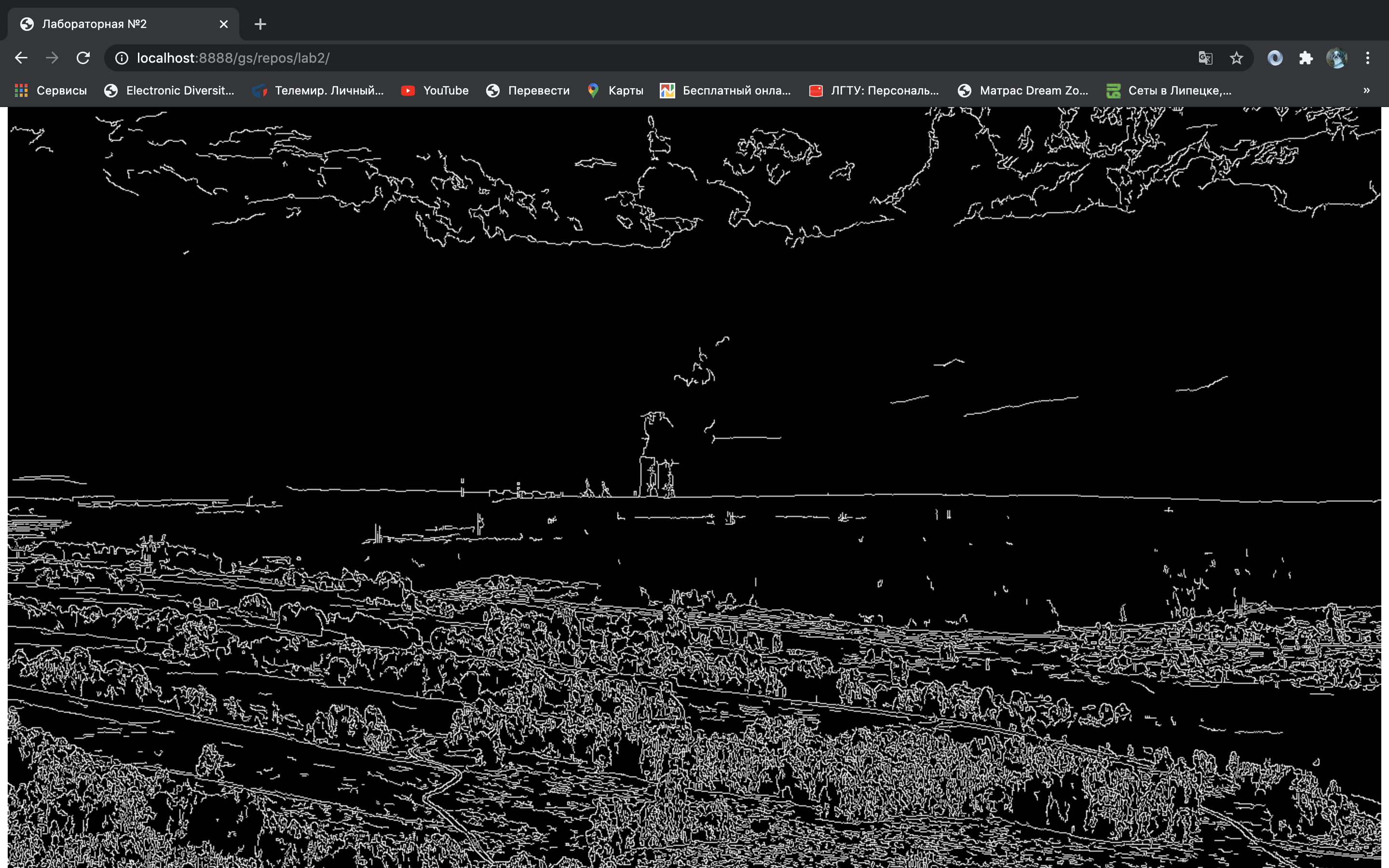


Рисунок 6 – Результат обработки

# Вывод

В ходе выполнения данной лабораторной работы была реализована программа определения границ объектов на изображениях при помощи алгоритма Канни. Разработанная программа использует вычислительные мощности графического процессора. Пользователь может загружать собственные изображения и обрабатывать их.