

Федеральное государственное автономное
образовательное учреждение
высшего образования
«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»
Институт Космических и информационных технологий
институт
Кафедра «Информатика»
кафедра

ОТЧЕТ О ПРАКТИЧЕСКОЙ РАБОТЕ

Матрицы приложение ИИ
тема

Преподаватель

подпись, дата

А.В. Пятаева
инициалы, фамилия

Студент КИ22-01-13М

номер группы, зачетной книжки

подпись, дата

Е.В. Железкин
инициалы, фамилия

Красноярск 2022

1 Цель работы

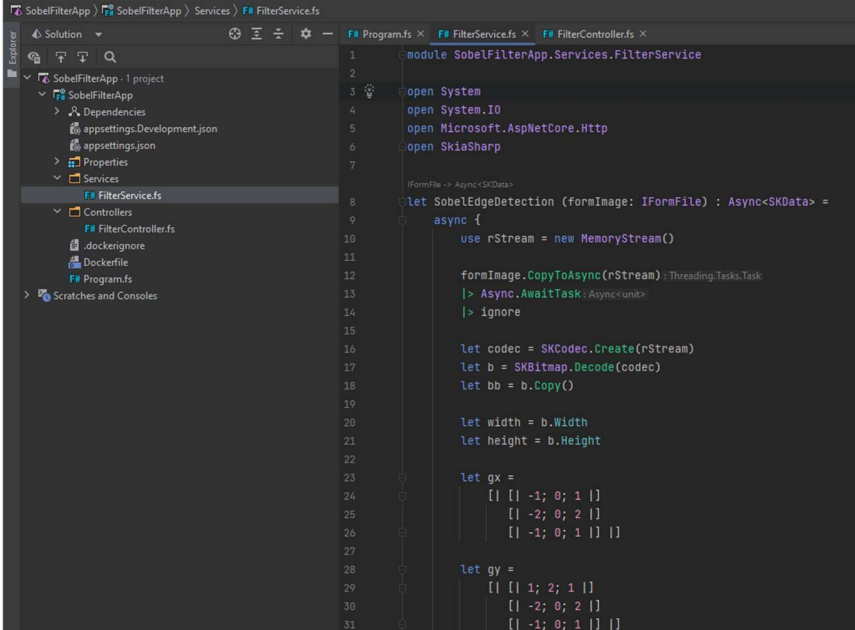
Изучить принципы матричной свёртки и создать приложение, применяющее фильтр Собеля к изображению.

2 Задача работы

Написать на любом языке программирования программу реализации операции свертки для матрицы (представляющей собой яркости пикселей) с фильтром Собеля.

3 Ход работы

Для простоты написания интерфейса, качестве решения было выбрано веб-приложение, обрабатывающее поступающее изображение. Программное решение разрабатывалось на платформе ASP.Net на языке F#. Интерфейс написан на стандартной html + css + js(jQuery).



```
1 module SobelFilterApp.Services.FilterService
2
3 open System
4 open System.IO
5 open Microsoft.AspNetCore.Http
6 open SkiaSharp
7
8 [FormFile -> Async<SKData>]
9 let SobelEdgeDetection (formImage: IFormFile) : Async<SKData> =
10     async {
11         use rStream = new MemoryStream()
12         formImage.CopyToAsync(rStream) :> Threading.Tasks.Task
13         > Async.AwaitTask : Async<unit>
14         > ignore
15
16         let codec = SKCodec.Create(rStream)
17         let b = SKBitmap.Decode(codec)
18         let bb = b.Copy()
19
20         let width = b.Width
21         let height = b.Height
22
23         let gx =
24             [ [ -1; 0; 1 ]
25               [ -2; 0; 2 ]
26               [ -1; 0; 1 ] ]
27
28         let gy =
29             [ [ 1; 2; 1 ]
30               [ -2; 0; 2 ]
31               [ -1; 0; 1 ] ]
```

Рисунок 1 – Код функции обработки изображения

```

F# Program.fs × F# FilterService.fs × F# FilterController.fs ×
24         [| [| -1; 0; 1 |]
25         [| -2; 0; 2 |]
26         [| -1; 0; 1 |] |]
27
28         let gy =
29             [| [| 1; 2; 1 |]
30             [| -2; 0; 2 |]
31             [| -1; 0; 1 |] |]
32
33         let allPixR =
34             Array2D.init width height (fun _ _ -> 0)
35
36         let allPixG =
37             Array2D.init width height (fun _ _ -> 0)
38
39         let allPixB =
40             Array2D.init width height (fun _ _ -> 0)
41
42         let limit = 128 * 128
43
44         for i in 0 .. width - 1 do
45             for j in 0 .. height - 1 do
46                 allPixR[i, j] <- Convert.ToInt32(b.GetPixel(i, j).Red)
47                 allPixG[i, j] <- Convert.ToInt32(b.GetPixel(i, j).Green)
48                 allPixB[i, j] <- Convert.ToInt32(b.GetPixel(i, j).Blue)
49
50         for i in 1 .. width - 2 do
51             for j in 1 .. height - 2 do
52                 let mutable newRx = 0
53                 let mutable newRy = 0
54                 let mutable newGx = 0
55                 let mutable newGy = 0
56                 let mutable newBx = 0
57                 let mutable newBy = 0
58
59                 for wi in -1 .. 1 do
60                     for hw in -1 .. 1 do
61                         let rc = allPixR[i + hw, j + wi]
62                         newRx <- gx[wi + 1][hw + 1] * rc
63                         newRy <- gy[wi + 1][hw + 1] * rc
64
65                         let gc = allPixG[i + hw, j + wi]
66                         newGx <- gx[wi + 1][hw + 1] * gc
67                         newGy <- gy[wi + 1][hw + 1] * gc
68
69                         let bc = allPixB[i + hw, j + wi]
70                         newBx <- gx[wi + 1][hw + 1] * bc
71                         newBy <- gy[wi + 1][hw + 1] * bc
72
73                 if newRx * newRx + newRy * newRy > limit
74                 || newGx * newGx + newGy * newGy > limit
75                 || newBx * newBx + newBy * newBy > limit then
76                     bb.SetPixel(i, j, SKColors.White)
77                 else
78                     bb.SetPixel(i, j, SKColors.Black)
79
80         return bb.Encode(SKEncodedImageFormat.Jpeg, 100)
81     }
82

```

Рисунок 2 – Код функции обработки изображения (продолжение)

```

F# Program.fs × F# FilterService.fs × F# FilterController.fs ×
1 namespace SobelFilterApp.Controllers
2
3 open System
4 open Microsoft.AspNetCore.Mvc
5 open SobelFilterApp.Services
6
7 [ApiController]
8 [Route("[controller]")]
9 type FilterController() =
10     inherit ControllerBase()
11
12     [HttpPost("[action]")]
13     member _.Sobel([<FromForm>] formImage) : Async<string> =
14         async {
15             let computedImageData =
16                 FilterService.SobelEdgeDetection formImage : Async<SkiaSharp.SKData>
17                 |> Async.RunSynchronously
18
19             return "data:image/jpeg;base64," + Convert.ToBase64String(computedImageData.ToArray())
20         }
21

```

Рисунок 3 – Endpoint для обработки изображения

```

Index.cshtml × script.js ×
1 @{
2     ViewData["Title"] = "Фильтр Собеля";
3 }
4 <script src="~/script.js" defer></script>
5
6 <div id="spinner" style="display: flex; align-items: center; justify-content: center;">
7     <div class="d-flex justify-content-center align-items-center" style="border: 2px solid #28a745; border-radius: 50%; width: 20px; height: 20px; margin: 0 auto 10px auto;">
8         <div class="spinner-border text-success" role="status"></div>
9     </div>
10 </div>
11 <div class="container">
12     <div class="row">
13         <div class="d-flex justify-content-center" style="margin-bottom: 10px;">
14             <h1>Фильтр Собеля</h1>
15         </div>
16     </div>
17     <div class="row d-flex justify-content-center" style="margin-bottom: 10px;">
18         <input id="file-input" type="file" class="form-control mb-2"/>
19         <div class="row d-flex justify-content-center">
20             <div id="preview-image-container"></div>
21         </div>
22         <button class="btn btn-outline-success mt-2" type="button" id="send-button">Обработать</button>
23     </div>
24     <div class="row d-flex justify-content-center" id="result-image-container"></div>
25 </div>

```

Рисунок 4 – Представление для веб страницы

```
Index.cshtml x script.js x
1 SaveFile = (file) => {
2     let formData = new FormData();
3     formData.append("formImage", file, file.name);
4
5     $.ajax({
6         type: "POST",
7         url: "/Filter/Sobel",
8         success: (data) => {
9             let imageContainer = $("#result-image-container");
10            let image = document.createElement('img');
11            image.setAttribute('src', data);
12            while (!imageContainer.empty()) {
13                imageContainer.remove(image.firstChild);
14            }
15            imageContainer.append(image);
16            $("#spinner").hide();
17        },
18        async: true,
19        data: formData,
20        cache: false,
21        contentType: false,
22        processData: false,
23        timeout: 60000
24    });
25 }
26
27 $("#send-button").on("click", function (_) {
28     $("#spinner").show();
29     let input = $("#file-input")[0];
30     let file = input.files[0];
31     if (file) {
32         SaveFile(file);
33     }
34 });
35
36 $("#file-input").on("change", function (e) {
37     if (e.target.files.length === 0) return;
38     let imageContainer = $("#preview-image-container");
39     let image = document.createElement('img');
40     image.setAttribute('id', 'image-preview');
41     image.setAttribute('style', 'width: 100%');
42     while (!imageContainer.empty()) {
43         imageContainer.remove(image.firstChild);
44     }
45     imageContainer.append(image);
46
47     let reader = new FileReader();
48     reader.onload = function (event) {
49         $('#image-preview').attr('src', event.target.result);
50     }
51     reader.readAsDataURL(e.target.files[0]);
52 });
53
54 $("#spinner").hide();
```

Рисунок 5 – Логика представления

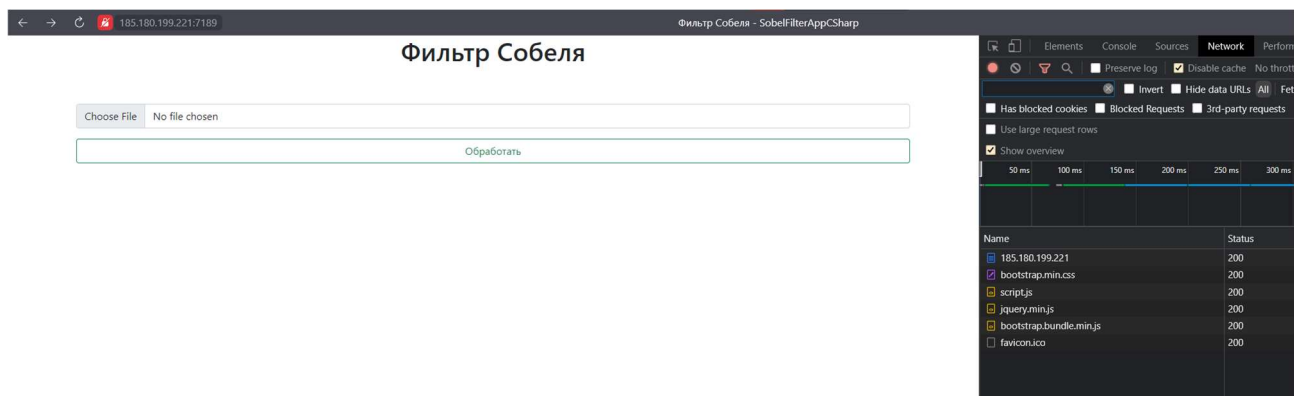


Рисунок 6 – Развёрнутое приложение



Рисунок 7 – Выбрана картинка для обработки

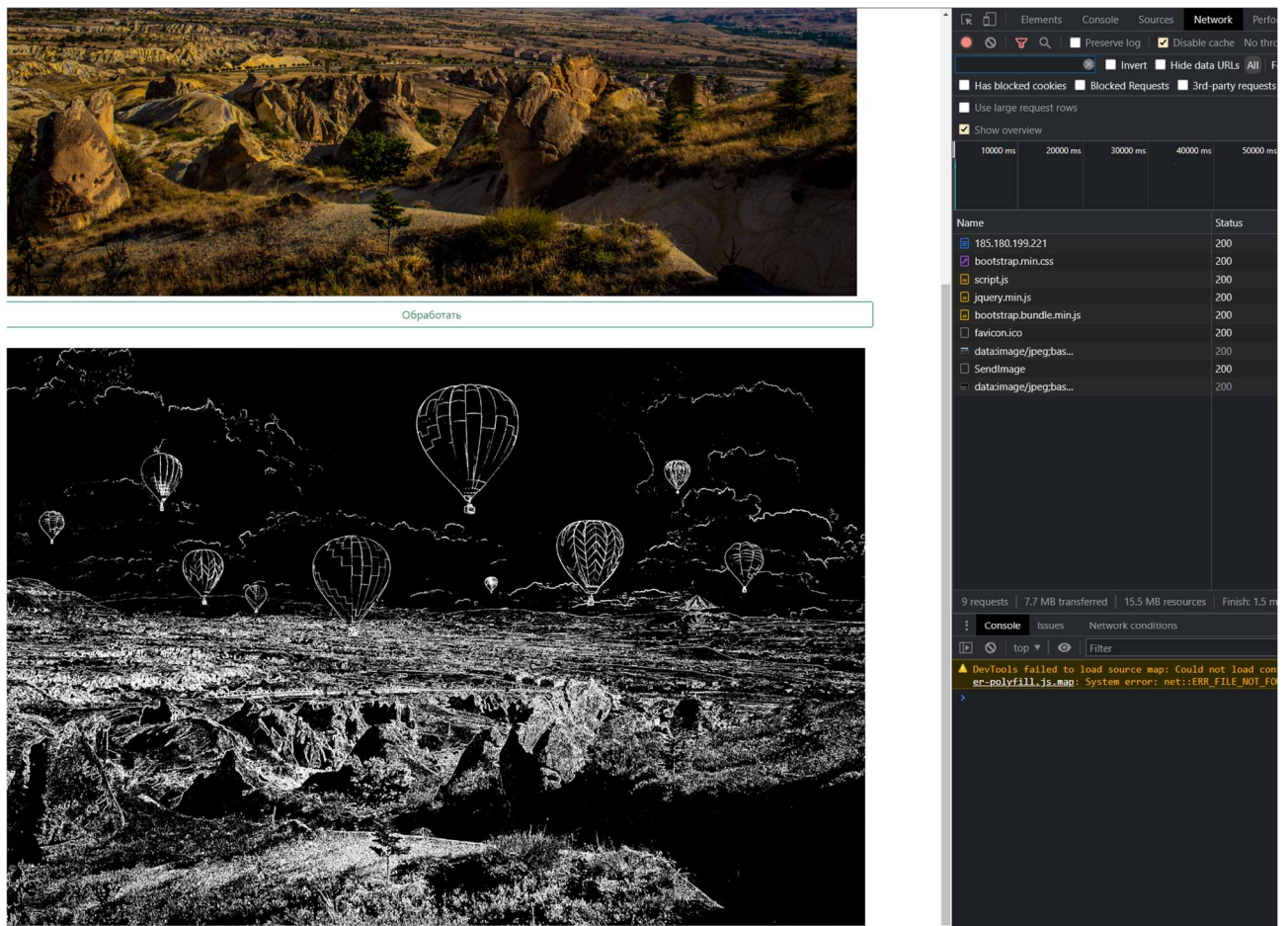


Рисунок 8 – Полученная картинка

Приложение доступно по адресу: <https://185.180.199.221:7189/>

Исходный код доступен по адресу: <https://github.com/Xorsiphus/SobelFilter-Univ>

4 Вывод

В ходе данной практической работы были изучены принципы матричной свёртки и создано приложение, применяющее фильтр Собеля к изображению.