Выделение контура на изображении

Изображение как матрица

Пусть дано цветное изображение (рис. 1) в виде файла с расширением `.jpg`.

Рис. 1: Исходное изображение

Для чтения графических файлов самым простым решением является библиотека `Pillow`.

```
#pip install pillow

from PIL import Image
image_RGB =Image.open(path)
```

С помощью `numpy` переведём изображение в матричный вид

```
import numpy as np

matrix_RGB = np.array(image_RGB)
print(matrix_RGB.shape)
```

```
(1280, 1280, 3)
```

Таким образом видно, что наше изображение имеет разрешение 1024×1024 пикселей, а каждому пикселю поставлено в соответствие три значения.

Цветное изображение на рисунке 1 можно представить в RGB-кодировке с помощью трёх матриц, отвечающих за красный (R), зелёный (G) и синий (B) цвета. Мы можем вывести отдельно эти слои, как показано на рисунке 2 (для наглядности использованы цветовые схемы `Reds`, `Greens`, `Blues`).

Пример кода для вывода матрицы “красных компонент”:

```
plt.imshow(matrix_R, cmap = 'Reds')
```

Рис. 2: Отдельные слои цветного изображения

Для простоты в дальнейшем будем работать не с цветным изображением, а чёрно-белым, которое задано в “оттенках серого” лишь одной матрицей. Для перевода существует эмпирическая формула, коэффициенты которой подобраны с учётом восприятия человеческого глаза

$$GS = 0.2989 \cdot R + 0.5870 \cdot G + 0.1140 \cdot B,$$

где R, G, B – значения красной, зелёной и синей компонент пикселя цветного изображения, GS – значение пикселя чёрно-белого изображения.

Получившееся после конвертации изображение показано на рисунке 3.

Рис. 3: Изображение в оттенках серого (`cmap = 'gist_gray'`)

Выделение контуров

Для полученного изображения мы можем смотреть вертикальные и горизонтальные срезы (рис. 4), наблюдая за изменением яркости по желаемому направлению. Для выделения контуров нас интересуют участки, где изменения яркости пикселей велики.

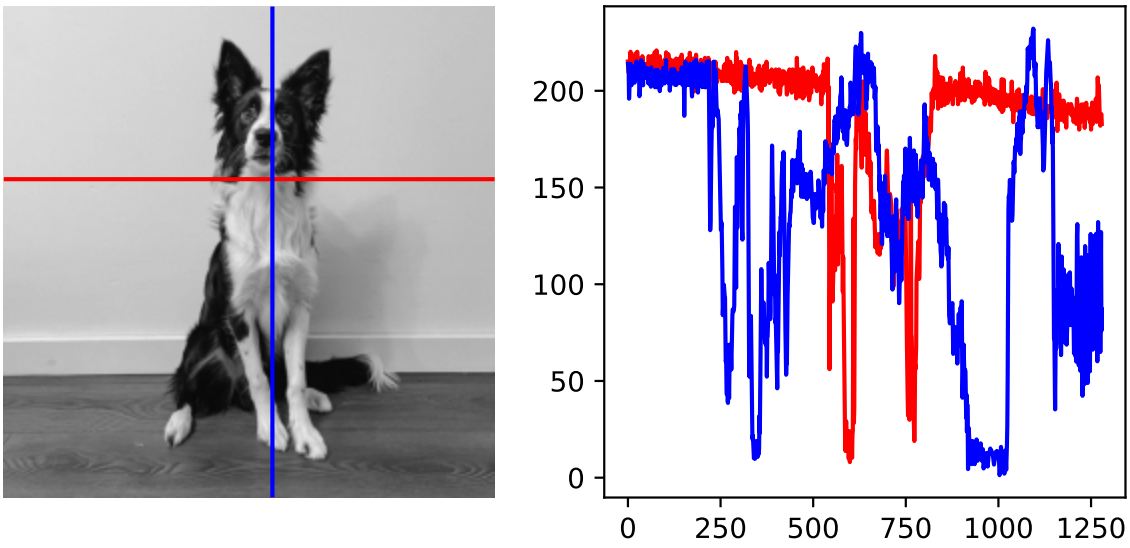

Рис. 4: Изображение и его срезы

Значения производных по вертикальному и горизонтальному направлениям найдём с помощью `np.gradient`.

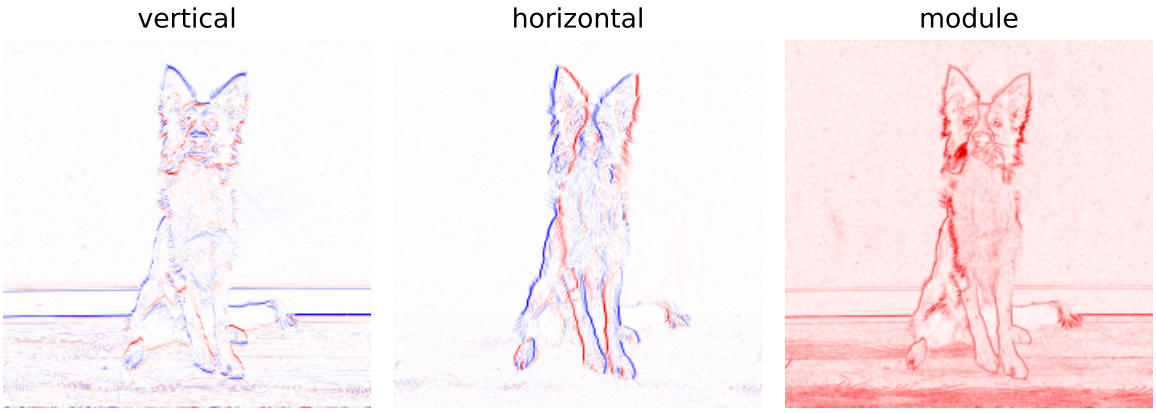

Рис. 5: Значения производных по направлениям и модуля градиента

Зададим пороговый уровень `threshhold` и бинаризуем изображение, составленное из модулей градиентов

$$I^*_{i,j} = \begin{cases} 0, & I_{i,j} < \texttt{threshhold}, \\ 1, & I_{i,j} \geq \texttt{threshhold}. \end{cases}$$

Для построения графика можно взять за основу код:

```
levels = np.linspace(10, 18, 9)

plt.figure(figsize = (8, 8))

for i in range(3):
    for j in range(3):
        matrix_threshold = np.where(matrix < levels[3*i+j], 0, 1)

        plt.subplot(3, 3, 3*i+j+1)
        plt.imshow(matrix_threshold, cmap='binary')
        plt.title(f'threshhold={np.round(levels[3*i+j],2)}')
        plt.axis('off')
```

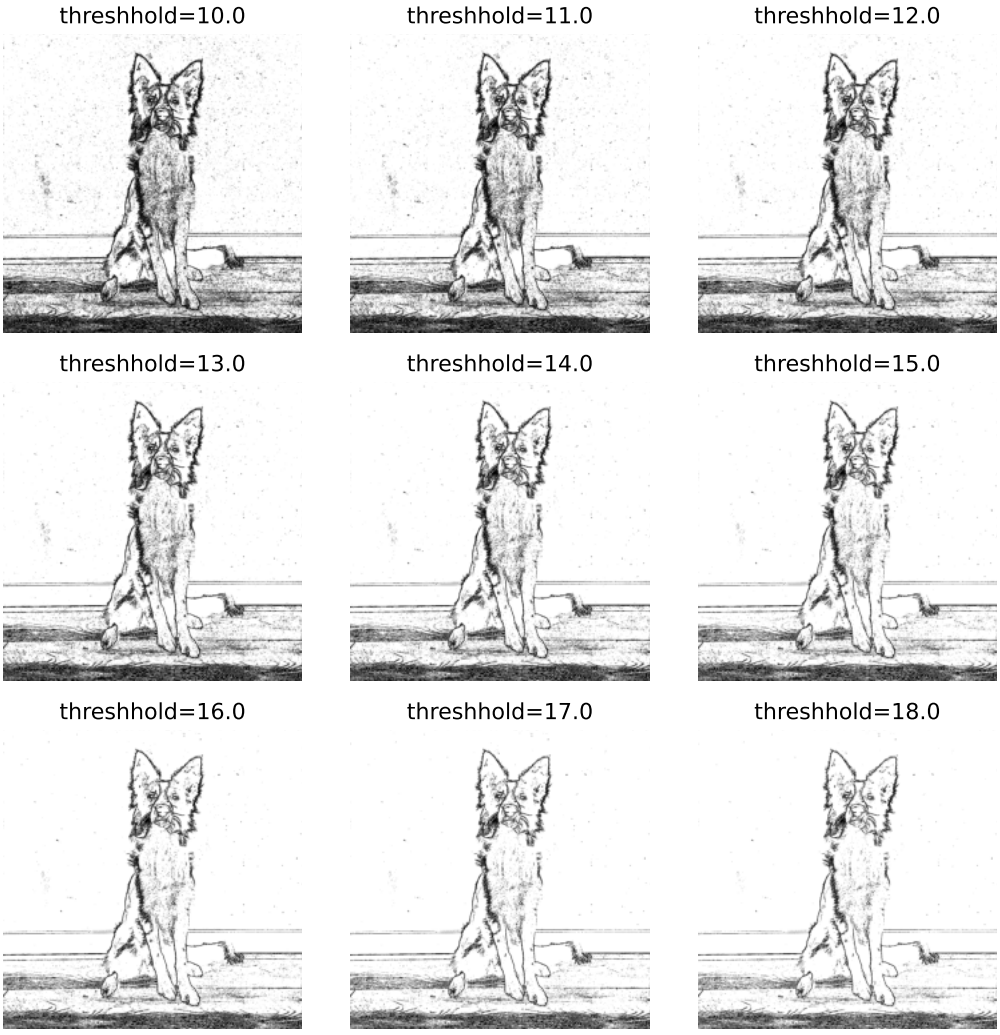

Рис. 6: Выделенные границы при различном уровне `threshhold`

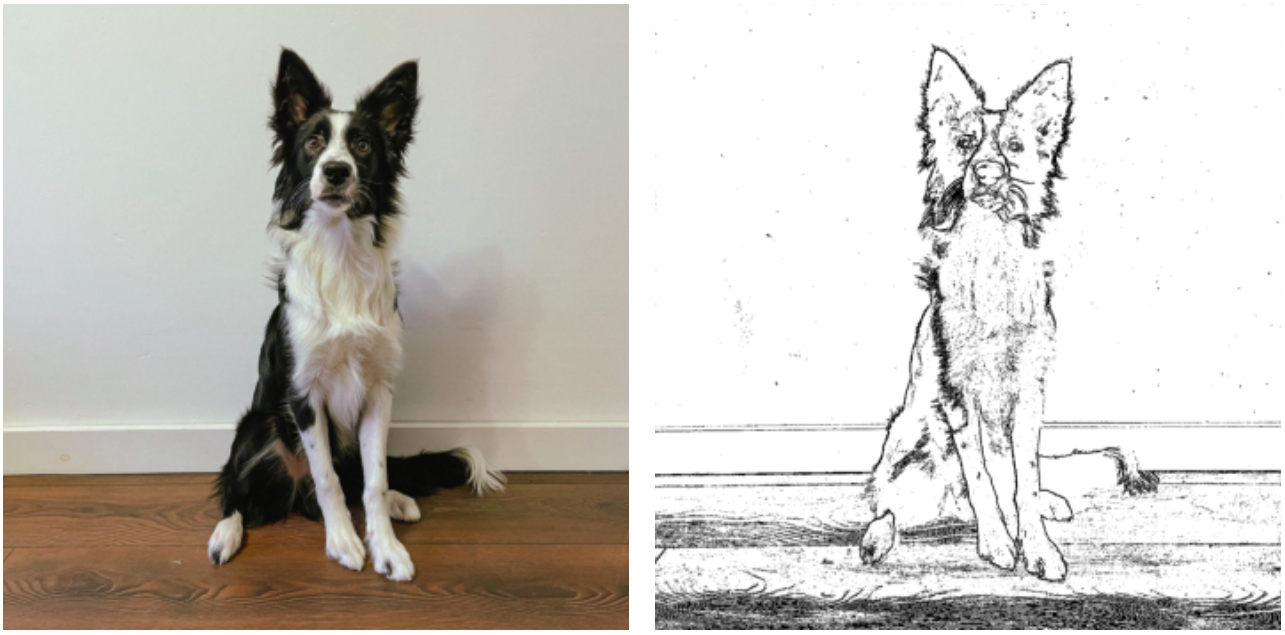

Рис. 7: Исходное изображение и выделенные границы

Задание

Необходимо реализовать все описанные шаги на своём изображении, построить рисунки аналогичные 1–7 и оформить отчёт.