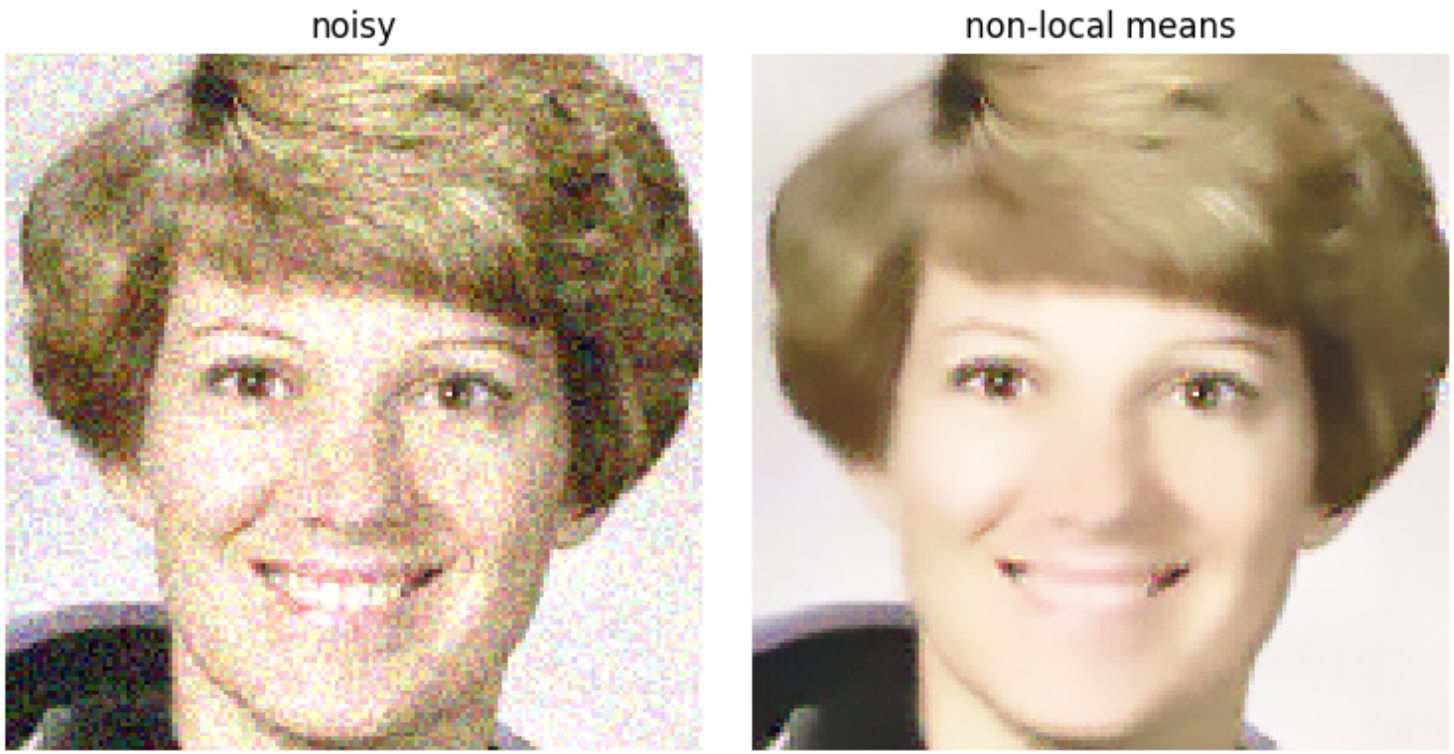
Занятие 5: Основы обработки изображений

Практикум на ЭВМ 2017/2018

Попов Артём Сергеевич

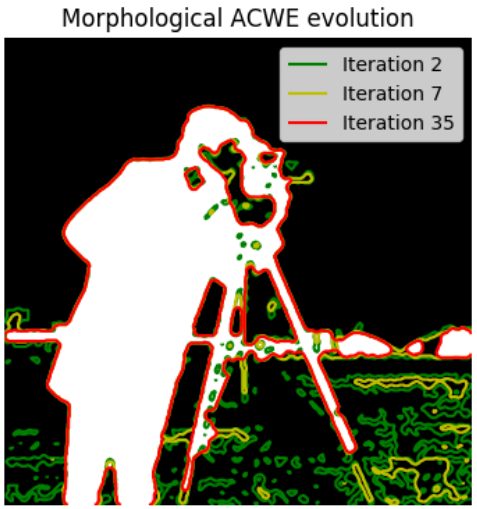
МГУ имени М. В. Ломоносова, факультет ВМК, кафедра ММП

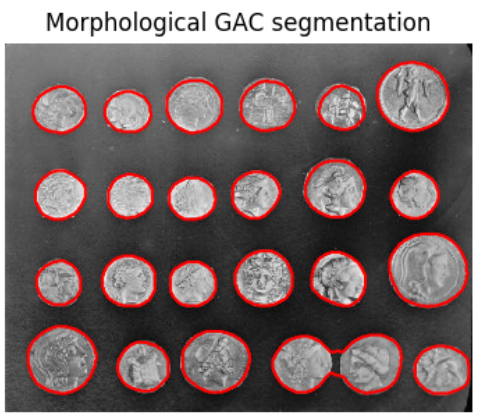
Пример: устранение шума

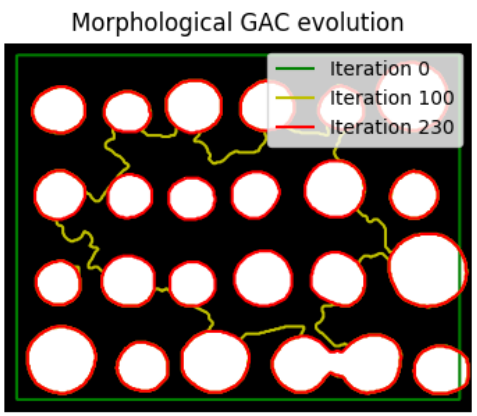


http://scikit-image.org/docs/dev/auto_examples/filters/...

Пример: выделение краёв







http://scikit-image.org/docs/dev/auto_examples/segmentation/...

Пример: изменение баланса цвета

http://studioeszkozok.hu/uploads/images/...

Зачем нужна обработка изображений?

1. Улучшение изображения для восприятия человеком (изображение должно стать «лучше» с субъективной точки зрения человека)
2. **Улучшение изображения для восприятия компьютером (улучшение качества работы алгоритмов)**
3. Технические нужды (например, уменьшение размера изображений для пересылки по почте)
4. Построение спецэффектов

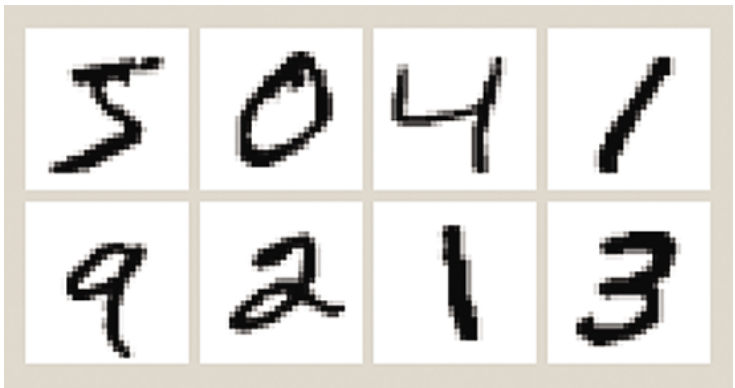
Обо всём этом подробнее в курсе «Компьютерная графика»

Улучшение качества работы алгоритма на изображениях

- Предобработка входных изображений
  - Удаление шума, преобразование цветов
- Выделение дополнительных признаков
  - Выделение важных объектов, областей
- Генерация дополнительных изображений для обучения
  - Добавление изображений, полученных из исходных с помощью некоторых преобразований
- Аугментация объектов
  - Преобразование объектов в ходе обучения/применения модели

Пример: классификация цифр

Первое практическое задание: классификация датасета MNIST



Класс изображения не меняется при:

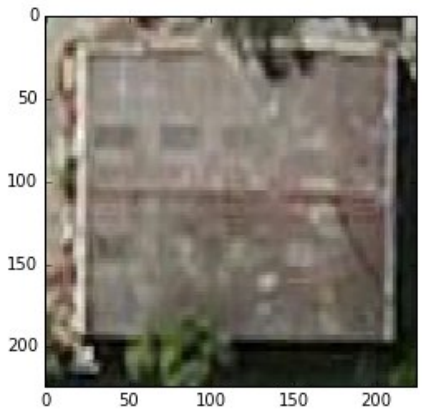
- сдвигах на 1-10 пикселей
- поворотах на 10-15 градусов в каждую из сторон
- размытии, удалении шумов

Добавив преобразованные объекты в исходную выборку, можно существенно повысить качество

Пример: классификация типов крыш

Задача: определение типа крыши, один из четырёх классов:

1. North-South orientation
2. East-West orientation
3. Flat roof
4. Other



1. При повороте на 90° объект 1 и 2 класса меняет класс.
2. При повороте на 180° объект 1 и 2 класса не меняет класс.
3. При повороте на 90° объект 3 класса не меняет класс.

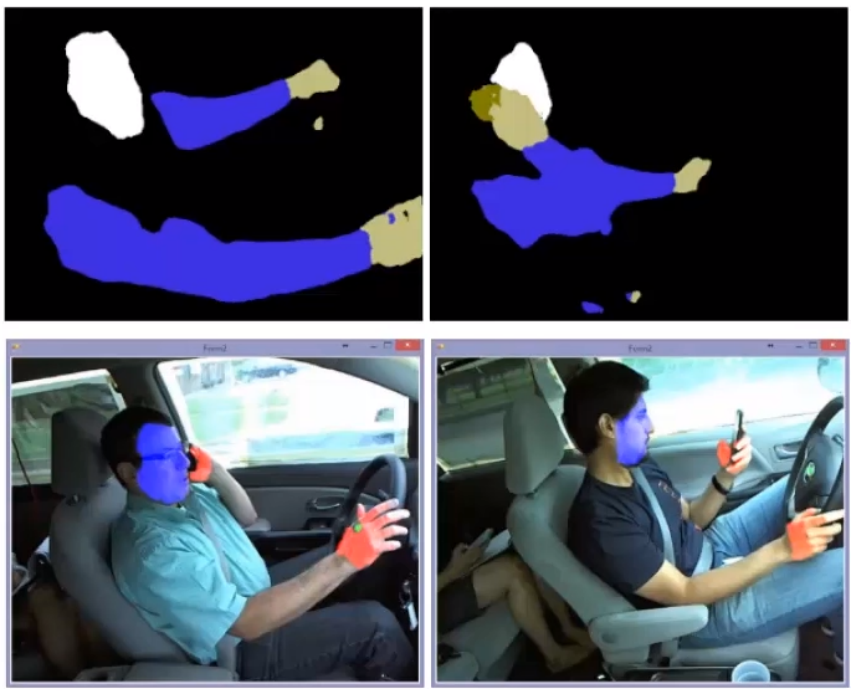
Аугментация объектов улучшила точность с 80% до 84%

Пример: детектирование действий водителя

Детектировать запрещённые действия водителей по видео:

Исходное изображение:

Выделение областей:



Алгоритм можно настраивать отдельно на выделенные области

Библиотека scipy

- Scipy — библиотека для научных вычислений https://scipy.org/
- В том числе, есть несколько модулей для работы с изображениями (misc, ndimage)
- Только самые базовые алгоритмы

Библиотека scikit-image

- Scikit-image — библиотека для работы с изображениями http://scikit-image.org/
- Большое количество реализованных алгоритмов для работы с изображениями
- Много выложенных примеров использования библиотеки, но плохая документация

Библиотека OpenCV

- OpenCV — продвинутая библиотека для компьютерного зрения, есть хороший интерфейс для Python 2.7 https://opencv-python-tutroals.readthedocs.io/en/latest/index.html
- Огромное количество реализованных алгоритмов для работы с изображениями, видео, 3D моделями
- Хорошая документация

Представление изображения в памяти (scipy)

```
>>> import matplotlib.pyplot as plt
>>> %matplotlib inline
>>> import scipy.misc as misc
...
>>> img = misc.imread('msu.png') # загрузка изображения
>>> print(type(img))  # изображение - трёхмерный numpy array
<class numpy.ndarray>
>>> print(img.shape)
(595, 800, 3)
>>> plt.imshow(img)
```

Представление изображения в памяти (skimage)

То же самое, но с scikit-image:

```
>>> import skimage.io as io
...
>>> img = io.imread('msu.png') # загрузка изображения
>>> print(type(img))  # изображение - трёхмерный numpy array
<class numpy.ndarray>
>>> print(img.shape)
(595, 800, 3)
>>> plt.imshow(img)
```

Сдвиги изображений

```
>>> import scipy.ndimage as ndimage
...
>>> new_img = ndimage.shift(img, [150, 150, 0])
>>> plt.imshow(new_img)
```

Повороты изображений (scipy)

```
>>> new_img = misc.imrotate(img, 90)
>>> plt.imshow(new_img)
```

Изображение обрезается, нельзя сохранить исходные пропорции, нельзя обрезать границы.

Повороты изображений (skimage)

```
>>> import skimage.transform as transform
>>> new_img = transform.rotate(img, 90, resize=True)
>>> plt.imshow(new_img)
```

Много параметров:

```
transform.rotate(image, angle, resize=False, center=None,
order=1, mode='constant', cval=0, clip=True,
reserve_range=False)
```

Оттенки серого

```
>>> import skimage.color as color
>>> grey_img = color.rgb2grey(img)
>>> plt.imshow(grey_img, cmap=plt.cm.Greys)
```

Удаление шумов

```
>>> noise_coords_i = np.random.randint(0, img.shape[0], 50000)
>>> noise_coords_j = np.random.randint(0, img.shape[1], 50000)
>>> img_copy = np.copy(grey_img)
>>> img_copy[noise_coords_i, noise_coords_j] = 0
...
>>> blur_img = filters.median(img_copy)
```

Заключение

- Используя методы обработки изображений, можно повысить качество работы алгоритма
- В Python есть несколько библиотек с уже реализованными алгоритмами обработки
- У многих библиотек есть свои встроенные средства обработки