ЛАБОРАТОРНАЯ РАБОТА №1
ИЗУЧЕНИЕ И ПРИМЕНЕНИЕ БАЗОВЫХ ОПЕРАЦИЙ ОБРАБОТКИ ИЗОБРАЖЕНИЙ

Цель занятия: ознакомление студентов с основными понятиями и навыками, необходимыми для эффективной работы с изображениями.

1.1 Scikit-image

Пакет для Питона, имеющий открытый исходный код и работающий с массивами NumPy. Позволяет реализовывать алгоритмы и утилиты с последующим их применением в научно-исследовательских, образовательных и даже промышленных программных приложениях. Пакет прост и понятен, с ним справятся даже новички. Код написан активным сообществом, реализован качественно и является рецензированным.

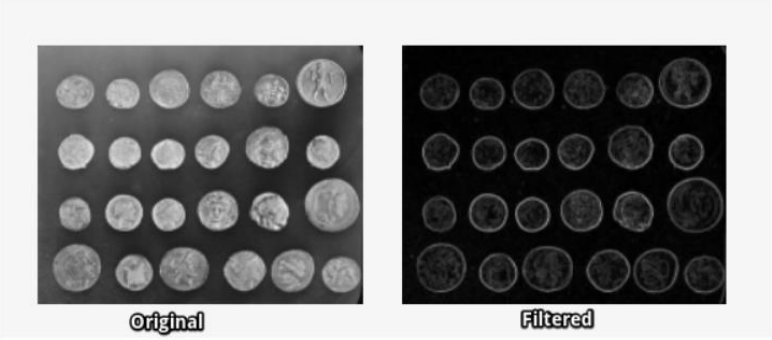
Также библиотека хорошо задокументирована, причем в документации можно найти много практических примеров.

Пакет импортируется как skimage, а большая часть функций располагается внутри подмодулей. На картинке ниже пример, каким путем можно выполнить фильтрацию изображений:

```
import matplotlib.pyplot as plt
 %matplotlib inline

from skimage import data,filters

image = data.coins()
 # … или любой массив NumPy!
 edges = filters.sobel(image)
 plt.imshow(edges, cmap='gray')
```



1.2 NumPy

Одна из основных и наиболее популярных Python-библиотек, которая поддерживает массивы. Изображение — это стандартный **NumPy**-массив, который содержит пиксели точек данных. В результате в процессе выполнения основных NumPy-операций (срезов, масок, индексирования) есть возможность менять пиксельные значения изображения. Непосредственно image можно загрузить с помощью skimage либо отобразить через Matplotlib.

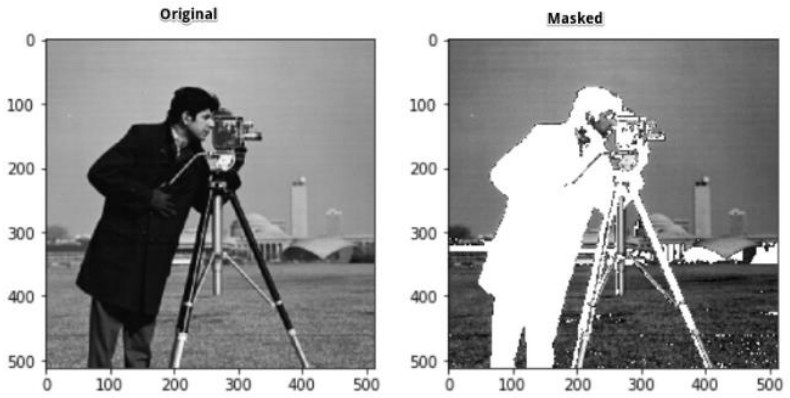
Все нужные ресурсы и документация есть на официальном сайте NumPy. Вот каким путем можно выполнить маскировку image с помощью NumPy:

```
import numpy as np
 from skimage import data
 import matplotlib.pyplot as plt
 %matplotlib inline

image = data.camera()
 type(image)

numpy.ndarray #Изображение - это массив NumPy

mask = image < 87
 image[mask]=255
 plt.imshow(image, cmap='gray')
```



Как видим, распознать внешность человека с камерой теперь невозможно.

1.3 SciPy

SciPy — не менее важный научный модуль, сравнимый с NumPy. Он позволяет решать основные задачи по обработке изображений и не только. К примеру, в подмодуле **scipy.ndimage** есть функции, работающие в n-мерных NumPy-массивах. Текущий пакет содержит функции для линейной/нелинейной фильтрации, интерполяции B-сплайнами, бинарной морфологии и пр. Полный список функций лучше смотреть в документации.

Рассмотрим, как SciPy используется для размытия изображений посредством фильтра Гаусса:

```
from scipy import misc,ndimage

face = misc.face()
blurred_face = ndimage.gaussian_filter(face, sigma=3)
very_blurred = ndimage.gaussian_filter(face, sigma=5)

#Результаты
plt.imshow(<image to be displayed>)
```



1.4 PIL, он же Pillow

PIL — это Python Imaging Library и бесплатная библиотека, предназначенная для открытия, работы, а также сохранения разных форматов изображений. Неприятная новость заключается в том, что ее разработка остановилась, причем последнее обновление выходило в далеком 2009. Но есть и хорошая новость: существует **Pillow**—форк PIL, который активно развивается и характеризуется простой установкой. Pillow работает на основных ОС и поддерживает Python 3-й версии. Имеет базовый функционал методов для обработки изображений, в том числе точечные операции, преобразование цветового пространства, фильтры с набором ядер свертки и пр.

Примеры использования и процесс установки хорошо описаны в документации.

Улучшаем image через ImageFilter в Pillow:

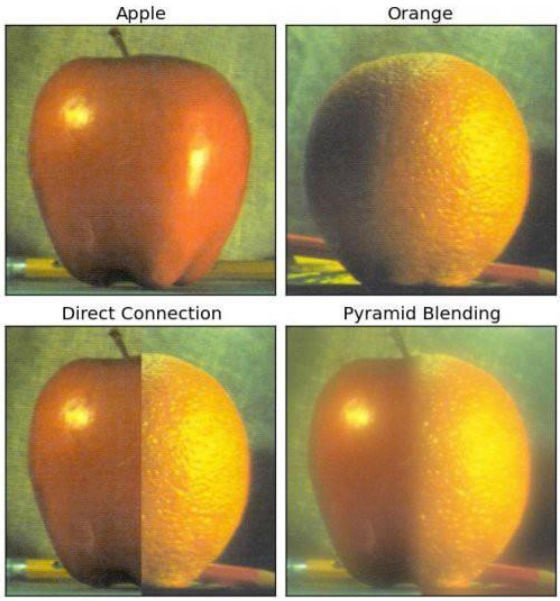
```
from PIL import Image, ImageFilter
#считывание изображения
im = Image.open( 'image.jpg' )
#отображение изображения
im.show()
from PIL import ImageEnhance
enh = ImageEnhance.Contrast(im)
enh.enhance(1.8).show("30% more contrast")
```



1.5 OpenCV

Одна из наиболее популярных библиотек, которая используется при обработке приложений по компьютерному зрению (**Computer Vision** — популярнейшее сегодня направление и технология, используемая в системах управления процессами, системах видеонаблюдения с камер, системах дополненной реальности и т. д.) Версия интерфейса для Пайтона называется **OpenCV-Python**, и именно Python-обертка во фронтенде обеспечивает легкое развертывание и настройку. Отличный инструмент для высоконагруженных вычислительных программ, работа которых связана с Computer Vision. Руководство здесь.

На jpg картинке ниже — пример наложения изображения посредством пирамид с целью создания нового фрукта.



1.6 SimpleCV

Очередной фреймворк с открытым исходным кодом, предназначенный для создания Computer Vision-приложений. С ним появляется доступ к ряду мощных CV-библиотек (к тому же OpenCV) без надобности изучения глубины цвета, цветовых пространств, файловых форматов и т. д. Кривая обучения гораздо меньше, чем в OpenCV, то есть основной слоган — «компьютерное зрение становится проще». Также среди преимуществ:

- простые CV-тесты смогут писать даже начинающие разработчики;
- алгоритм работы обеспечивает функциональную совместимость не только с изображениями (images), но и с камерой (camera), видеофайлами, видеопотоками.

А вот и понятная документация, где можно найти много практических примеров. Один из них — на фотографии ниже:



1.7 Mahotas

Очередная библиотека Python для обработки изображений и Computer Vision. Включает в себя как стандартные обрабатывающие функции (фильтрация, морфологические операции), так и более современные CV-возможности, позволяющие

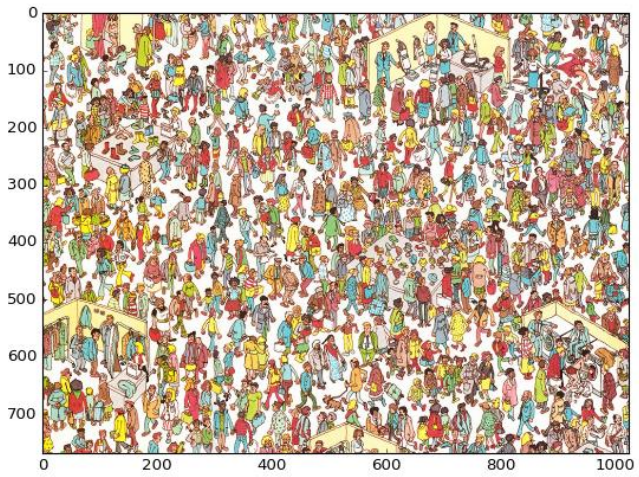
вычислять признаки и находить особые точки и локальные дескрипторы. Python-интерфейс обеспечивает высокую скорость разработки, а алгоритмы написаны на C++. В результате Mahotas представляет собой быструю библиотеку с минимальным кодом и зависимостями. В документации по ссылке можно найти инструкции по установке и даже пошаговые уроки по освоению.

Решение задач по поиску и обработке происходит с помощью простого кода. На картинке jpg ниже можно увидеть задачу по поиску под названием «Где Уолли?». Для выполнения поиска требуется минимум кода.

The problem is to find Wally (who goes by Waldo in the US) in the following image:

```
from pylab import imshow, show
import mahotas
import mahotas.demos
wally = mahotas.demos.load('Wally')
imshow(wally)
show()
```

(Source code, png, hires.png, pdf)



На рисунке ниже — результат поиска:

