Практикум-1 (Компьютерное зрение, 2014)

Вахитов А.Т.

March 6, 2014

1 Освоение Python

Для настройки системы необходимо следовать шагам инструкции Install OpenCV-Python in Windows (`http://docs.opencv.org/trunk/doc/py_tutorials/py_setup/py_setup_in_windows/py_setup_in_windows.html`) на сайте docs.opencv.org. Необходимо следовать инструкции до конца, установив и numpy, и matplotlib.

После установки, для освоения базовых конструкций языка, необходимо прочитать руководство по языку, озаглавленное The Python Tutorial, на сайте docs.python.org, при этом выбрать версию Python 2.7.6 вверху экрана `http://docs.python.org/2/tutorial/`. Для базового понимания конструкций языка, требуется прочесть главы 1-5 The Python Tutorial. Желательно не просто читать, а выполнять упражнения, предлагаемые в тексте.

Далее, требуется осовить базовые операции с изображениями с помощью библиотеки OpenCV. Для этого нужно прочитать раздел Basic operations on Images из OpenCV-Python Tutorials (`http://docs.opencv.org/trunk/doc/py_tutorials/py_tutorials.html`). Вы узнаете о том, как менть значения пикселей изображения, считывать их командой cv2.imread и записывать - командой cv2.imwrite.

1.1 Нотация

Согласуясь с Python, будем индексировать изображения, начиная с 0, так что левый верхний пиксель имеет координату (0,0), далее первая координата соответствует движению от левого верхнего угла вниз, а вторая координата - движению вправо.

1.2 Полезные команды

В начале любой программы будем использовать следующие модули:

```
import cv2
import numpy as np
import matplotlib.pyplot as plot
```

Для хранения изображения используется array из библиотеки numpy. Чтобы создать одноканальное изображение, необходимо выполнить команду

```
np.zeros( (imside_v, imside_h), dtype = ('u1'))
```

Здесь np - обращение к пространству имен библиотеки numpy, imside_v - вертикальный размер изображения, imside_h - горизонтальный размер изображения, u1 - код для обозначения того факта, что изображение будет содержать однобайтовый беззнаковые целые числа (стандартное представление для изображения в OpenCV).

2 Частотная фильтрация

Для начала, необходимо закрепить понимание спектрального анализа сигналов. Для этого предназначены упражнения из текущего раздела.

2.1 Построение графика спектра

Необходимо считать изображение chessboard.bmp (команда cv2.imread), выбрать пиксели строки 4 (то есть пятой строки изображения) и выполнить одномерное дискретное преобразование Фурье (ДПФ) над полученным вектором командой np.fft.fft. Затем посчитайте вектор модулей компонент ДПФ. Нужно отобразить этот вектор в виде графика командами

```
plot.plot(magnitude_spectrum, linewidth = 2.0);
plot.show();
```

Здесь magnitude_spectrum - это вектор модулей компонент ДПФ.

Объясните, пожалуйста, форму графика (симметричность или несимметричность, наличие нулевых/ненулевых компонент), используя формулу ДПФ:

$$X[n] = \sum_{k=0}^{N-1} x[k] W_n^k,\ W_n = e^{-j\frac{2\pi n}{N}},$$

где X - дискретное преобразование Фурье вектора x длины N.

Постройте изображение модулей компонент ДПФ от всего изображения chessboard.bmp размером 32 на 32 с помощью np.fft.fft2, отобразите его с помощью команд

```
plot.imshow(magnitude_spectrum);
plot.show();
```

Здесь magnitude_spectrum - это вектор модулей компонент двумерного ДПФ от всего изображения.

Объясните это изображение модулей компонент ДПФ.

Сохраните полученные график и двумерную карту в файлы fig1.bmp, fig2.bmp.

3 Изменение размера изображения

3.1 Дискретизация сигнала

Дискретизация сигнала $x_c(t)$ задается известным равенством

$$x[n] = x_c(nT_s),$$

где T_s - период, т.е. расстояние между отсчетами.

3.2 Реконструкция непрерывного сигнала по дискретному

Для реконструкции непрерывного сигнала без потерь используется идеальный фильтр нижних частот:

$$y(t) = \sum_{n=-\infty}^{\infty} \frac{\sin(\Omega_c t - n\pi)}{\pi(t - nT_s)} x[n],\ \Omega_c = \frac{\pi}{T_s}.$$

Однако, при практической реализации фильтра бесконечная сумма приближается конечным числом слагаемых. Для формальной записи этого факта существует оконная функция:

$$\bar{y}(t) = \sum_{n=-\infty}^{\infty} \frac{\sin(\Omega_c t - n\pi)}{\pi(t - nT_s)} x[n] w[t, n],$$

$$w[t, n] = \begin{cases} 1, |t - nT_s| < M/2 \\ 0, |t - nT_s| \geq M/2. \end{cases}$$

число M задает количество членов в приближении, чем оно выше, тем ближе результат к идеально реконструированному сигналу. В практической реализации фильтра, не обязательно вычислять оконну функцию так, как написано в формуле - для каждого элемента массива x.

3.3 Фильтрация нижних частот идеальным фильтром

Для удаления высоких частот из сигнала используется фильтр нижних частот с импульсной реакцией:

$$h[n] = \frac{\sin(\omega_c(n - M/2))}{\pi(n - M/2)} w[n],$$

$$w[n] = \begin{cases} 0.5 - 0.5\cos(2\pi n/M), & 0 \leq n \leq M, \\ 0, & \text{иначе} \end{cases}$$

Уменьшение строки изображения с потерями. Масштабирование изображения, то есть изменение его размера, можно воспринимать как последовательность двух шагов: реконструкции непрерывного сигнала и дискретизации с новым шагом, более частым или более редким, чем исходный. Необходимо взять строку 4 (т.е. пятую) в изображении chessboard.bmp, реконструировать по ней неперывный сигнал, считая $T_s = 1/32$, $M = 4$, затем взять новый $T_{s'} = 1/8$ и произвести дискретизацию, на

выходе должно получиться 8 элементов. Полученный массив отрисовать в виде графика командой *plot.plot* и сохранить в fig2_1.bmp. Для крайних значений, где окно выходит за пределы исходного сигнала, будем циклически продолжать окно, начиная с противоположного края. Мы считаем, что сигнал бесконечный и однородный, поэтому используем циклическое окно. Иными словами, мы продолжим исходный сигнал на $M/2$ влево и вправо, и будем реконструировать уже его.

Далее, по сигналу из 8 компонент, необходимо реконструировать непрерывный сигнал и построить его дискретизацию с периодом $T_s = 1/32$. Полученный массив сравнить с исходным, т.е. строкой 4 изображения chessboard.bmp, посчитать норму разности между ними. Построить график полученного массива, сохранить в fig2_2.bmp. Норму разности вывести в файл norm_row.txt.

Уменьшение строки изображения без потерь. Отфильтровать строку 4 изображения chessboard.bmp идеальным фильтром нижних частот с частотой $\pi\frac{T_s}{T_{s'}} = \pi/4$. Затем проделать операции, аналогичные описанным в предыдущем пункте, но для отфильтрованной строки. Сохранить графики в fig3_1.bmp, fig3_2.bmp. Норму разности меджу отфильтрованной строкой и реконструированной строкой вывести в norm_row_filt.txt. Объяснить графики и численные результаты этих пунктов.

Уменьшение всего изрображения с потерями и без потерь. Произвести действия, аналогичные изложенным в предшествующих пунктах, но для всего изображения chessboard.bmp, используя двумерный фильтр низких частот. Вместо графиков выводить карты (как описано ранее, командой plot.imshow()). Сохранить карты в fig4_1.bmp, fig4_2.bmp и fig5_1.bmp, fig5_2.bmp. Сохранить норму в norm_image.txt, norm_image_filt.txt.