ЛАБОРАТОРНАЯ РАБОТА №3: ШУМОПОДАВЛЕНИЕ И ФИЛЬТРАЦИЯ

Цель работы: Изучить методы шумоподавления и фильтрации биомедицинских изображений с использованием Python и библиотек для обработки изображений.

В области обработки изображений шумоподавление и фильтрация играют ключевую роль в улучшении качества изображений, особенно в биомедицинских приложениях, где точность и четкость данных критичны для диагностики и анализа.

Шумоподавление направлено на удаление нежелательных случайных вариаций в изображениях, которые могут искажать информацию. Одним из наиболее распространённых видов шума в медицинских изображениях является гауссов шум, который добавляется в результате различных процессов, таких как сенсорные помехи или артефакты.

Для шумоподавления применяются различные **фильтры**:

1) **Средний фильтр (Mean filter):** Заменяет значение пикселя на среднее значение всех пикселей в окне вокруг него. Формально, для пикселя $I(i,j)$ в окне размером $m \times n$:

$$I_{new}(i,j) = \frac{1}{m \cdot n} \sum_{k=-m/2}^{m/2} \sum_{l=n/2}^{n/2} I(i+k, j+l)$$

2) **Медианный фильтр (Median filter):** Заменяет значение пикселя на медиану всех пикселей в окне вокруг него. Это эффективный способ удаления шума, сохраняя края:

$$I_{new}(i,j) = median\{I(i+k, j+l)\}$$

3) **Гауссов фильтр (Gaussian filter):** Использует гауссово распределение для взвешивания пикселей в окне. Формула для гауссового фильтра выглядит следующим образом:

$$G(x,y) = \frac{1}{2\pi\sigma^2} exp\left(-\frac{x^2+y^2}{2\sigma^2}\right)$$

где σ – стандартное отклонение гауссового распределения

4) **Билатеральный фильтр (Bilateral filter):** Сохраняет края, учитывая как пространственную близость, так и разницу в значениях интенсивности:

$$I_{new}(i,j) = \frac{1}{W} \sum_{k=-m/2}^{m/2} \sum_{l=-\frac{n}{2}}^{\frac{n}{2}} I(i+k, j+l) \cdot G_{spatial}(k,l) \cdot$$
$$\cdot G_{range}(I(i+k, j+l) - I(i,j))$$

где $G_{spatial}$ и G_{range} – функции гауссового распределения для пространственной близости и разности интенсивностей соответственно.

Оценка качества фильтрации. Для оценки качества фильтрации изображений применяются различные метрики, такие как ***PSNR*** **(Peak Signal-to-Noise Ratio)** и ***SSIM*** **(Structural Similarity Index)**. Эти метрики позволяют количественно оценить, насколько восстановленное изображение похоже на оригинальное.

PSNR **(Peak Signal-to-Noise Ratio)** измеряет уровень шума по отношению к сигналу и часто используется для оценки качества изображений. Это отношение максимальной возможной мощности сигнала к мощности шума. Формально *PSNR* вычисляется следующим образом:

$$PSNR = 10\lg(\frac{R^2}{MSE})$$

где R – максимальное значение пикселя (обычно 255 для 8-битных изображений), а *MSE* – среднеквадратичная ошибка (Mean Squared Error):

$$MSE = \frac{1}{m \cdot n}\sum_{i=1}^{m}\sum_{j=1}^{n}[I(i,j) - K(i,j)]^2$$

где *I(i,j)* и *K(i,j)* – значения пикселей в оригинальном и восстановленном изображениях соответственно, а m и n – размеры изображения.

SSIM **(Structural Similarity Index)** предназначен для оценки визуального качества, учитывая изменение структуры, контраста и яркости изображения. *SSIM* оценит схожесть между двумя изображениями в более комплексном контексте, чем *PSNR*. Формула *SSIM* следующая:

$$SSIM(x,y) = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)}$$

где:

1) μ_x и μ_y – средние значения пикселей в изображениях *x* и *y*,
2) σ_x^2 и σ_y^2 – дисперсии пикселей в изображениях *x* и *y*,
3) σ_{xy} – ковариация между изображениями *x* и *y*,
4) C_1 и C_2 – константы, предотвращающие деление на ноль (обычно $C_1=0.01^2 \cdot L^2$ и $C_2=0.03^2 \cdot L^2$, где L – динамический диапазон пикселей).

PSNR и *SSIM* предоставляют различные перспективы для оценки качества изображений, где *PSNR* фокусируется на уровне шума и максимальной мощности сигнала, в то время как *SSIM* учитывает визуальные и структурные изменения, обеспечивая более человеческий подход к оценке качества.

Задание 1. Загрузка и визуализация изображения

1. Импортируйте необходимые библиотеки (‘numpy’, ‘matplotlib’, ‘scikit-image’).

2. Загрузите биомедицинское изображение с использованием функции ‘skimage.io.imread’.

3. Визуализируйте оригинальное изображение с помощью matplotlib.pyplot.imshow.
Вариант 1: Загрузите изображение из предоставленного файла и визуализируйте его зеленую компоненту.
Вариант 2: Загрузите изображение из предоставленного файла и визуализируйте его красную компоненту.
Вариант 3: Загрузите изображение из предоставленного файла, преобразуйте его в оттенки серого и визуализируйте.
Вариант 4: Загрузите изображение из предоставленного файла, затем используйте метод ‘skimage.color.rgb2gray’ для преобразования цветного изображения в градации серого перед визуализацией.
Вариант 5: Загрузите изображение из предоставленного файла и визуализируйте его синюю компоненту.
Вариант 6: Загрузите изображение из предоставленного файла и визуализируйте сумму синей и красной компонент.

Задание 2. Добавление шума
1. Добавьте гауссов шум к изображению с помощью функции ‘skimage.util.random_noise’ с параметром ‘mode='gaussian''.
2. Визуализируйте изображение с шумом.
Вариант 1: Используйте стандартное значение параметра *var* для гауссового шума.
Вариант 2: Экспериментируйте с различными значениями параметра *var* в диапазоне от 0.01 до 0.1 с шагом 0.01 и визуализируйте результаты.
Вариант 3: Экспериментируйте с различными значениями параметра *var* в диапазоне от 0.1 до 0.5 с шагом 0.1 и визуализируйте результаты.
Вариант 4: Экспериментируйте с различными значениями параметра *var* в диапазоне от 0.001 до 0.005 с шагом 0.001 и визуализируйте результаты.
Вариант 5: Экспериментируйте с различными значениями параметра *var* в диапазоне от 0.5 до 1.0 с шагом 0.1 и визуализируйте результаты.
Вариант 6: Экспериментируйте с различными значениями параметра *var* в диапазоне от 0.0001 до 0.001 с шагом 0.0001 и визуализируйте результаты.

Задание 3. Применение фильтров
1. Примените средний фильтр (Mean filter) к изображению с шумом с помощью функции ‘skimage.filters.rank.mean’ из модуля ‘skimage.filters’.
2. Примените медианный фильтр (Median filter) с помощью функции ‘skimage.filters.median’.
3. Визуализируйте результаты применения фильтров.

Вариант 1: Используйте фильтры с квадратным окном размера 3x3.
Вариант 2: Используйте фильтры с квадратным окном размера 5x5 и сравните результаты.
Вариант 3: Используйте фильтры с прямоугольным окном размера 1x5.
Вариант 4: Используйте фильтры с прямоугольным окном размера 5x1.
Вариант 5: Используйте фильтры с окном круг размера 3.
Вариант 6: Используйте фильтры с окном круг размера 5.

Задание 4. Сравнение результатов

1. Создайте графики, показывающие оригинальное изображение, изображение с шумом и результаты фильтрации.
2. Используйте ‘matplotlib.pyplot.subplots’ для создания сравнительных графиков.
3. Сравните только средний фильтр и медианный фильтр.
4. Примите гауссовый фильтр и добавьте в графики результаты его применения.

Задание 5. Сравнение методов фильтрации

1. Примените два метода фильтрации для удаления гауссового шума: гауссов фильтр и фильтр Билатеральный фильтр. Используйте ‘skimage.filters.gaussian’ для гауссового фильтра и ‘skimage.restoration.denoise_bilateral’ для фильтра Билатерального фильтра.
2. Визуализируйте результаты фильтрации.

Задание 6. Оценка качества фильтрации

1. Рассчитайте *PSNR* (Peak Signal-to-Noise Ratio) и *SSIM* (Structural Similarity Index) между оригинальным изображением и изображениями после фильтрации.
2. Используйте функции ‘skimage.metrics.peak_signal_noise_ratio’ и ‘skimage.metrics.structural_similarity’ для вычисления этих показателей.
3. Рассчитайте *PSNR* и *SSIM* для одного метода фильтрации.
4. Сравните *PSNR* и *SSIM* для всех применённых методов фильтрации.
5. Оцените *PSNR* и *SSIM* для изображений с различными уровнями шума.
6. Постройте графики зависимости *PSNR* и *SSIM* от уровня шума.