FICH, UNL - Departamento de Informática - Ingeniería Informática

Procesamiento Digital de Imágenes

Guía de Trabajos Prácticos 6

Restauración de imágenes

1 Objetivos

- Comprender la dinámica de los modelos de ruido y sus parámetros descriptivos.
- Formular estrategias para el filtrado de ruido en el dominio espacial y frecuencial.
- Experimentar con el desempeño de las diversas opciones de filtros de medias y de orden, y sus efectos en cascada.
- Estudiar los efectos de restauración que realizan los filtros de deconvolución.

2 Trabajos Prácticos

Para esta guía le serán **muy útiles las implementaciones previas**, sobre todo las relacionadas a la convolución espacial, el cálculo de histogramas y el filtrado frecuencial.
Antes de comenzar, le recomendamos estudiar el funcionamiento de las siguientes funciones de openCV (`https://docs.opencv.org/`), numpy (`https://numpy.org/`):

- `dst = cv.randn(dst, mean, stddev)`
  versión numpy: `noise = np.random.normal(mean, sigma, (row,col))`
  recuerde siempre recuperar el tamaño de la imagen:
  `noise_img = noise.reshape(row,col)`
- `dst = cv.randu(dst, low, high)`
  versión numpy: `noise = np.random.rand(row,col)`
- Cómputo del ECM (MSE, mean square error)

```
def mse(imageA, imageB):
   err = np.sum((imageA.astype("float") - imageB.astype("float")) ** 2)
   err /= float(imageA.shape[0] * imageA.shape[1])
   return err
```

  o en bien en
  `https://docs.opencv.org/master/da/d0b/namespacecv_1_1quality.html`
  o
  `https://scikit-image.org/docs/dev/api/skimage.measure.html`

Ejercicio 1: Modelos de ruido

1. Genere imágenes utilizando los diferentes modelos de ruido: normal, uniforme, sal y pimienta, impulsivo unimodal y exponencial. etc.).
   Estudie la distribución obtenida analizando el histograma.
   Adapte sus funciones para que los ruidos gaussiano, uniforme y exponencial tengan su media en 0 (cero).

2. Genere una imagen (600×600px) que contenga 3 franjas verticales (600×200 px) de grises constantes, utilizando un gris claro (≈ 180), uno medio (≈ 120) y uno oscuro (≈ 60), y visualice el histograma.
   Sume los ruidos generado previamente, con valor medio en cero, y vizualice los histogramas.
   Varíe los parámetros del ruido (media, desvío, etc.) y verifique los efectos en el histograma para cada porción de grises constantes.

Ejercicio 2: Filtros de medias

1. Implemente los filtros de la media geométrica y de la media contra-armónica.

2. Genere una imagen ruidosa a partir de `'sangre.jpg'`, contaminándola con mezcla de ruido impulsivo y gaussiano.

3. Aplique los filtros y verifique la restauración de forma cualitativa. Luego, evalue los resultados cuantitativamente mediante la comparación del ECM (entre cada imagen filtrada y la limpia) vs. el ECM (entre la imagen degradada y la limpia).

Ejercicio 3: Filtros de orden

1. Implemente los siguientes filtros y aplíquelos a la misma imagen degradada del ejercicio anterior.
   (a) Filtro de mediana,
   (b) Filtro del punto medio,
   (c) Filtro de media-alfa recortado.
   (d) Aplique sucesivamente el filtro (a) y luego el filtro (b).

2. Indique en cuál de los casos se logra una mejor remoción del ruido.
   ¿Qué particularidades observa en cada uno de los resultados?
   Visualice los histogramas antes y después de cada uno de los filtrados.
   Compare los resultados subjetivamente y mediante el ECM.

3. Compare, discuta y saque conclusiones respecto de los resultados del filtrado de medias del Ejercicio 2.

Ejercicio 4: (Opcional) Filtro adaptativo de reducción local del ruido

$$\hat{f}(x,y) = g(x,y) - \frac{\sigma_\eta^2}{\sigma_L^2}\left[g(x,y) - m_L\right]$$

1. Cargue una imagen y realice un degradado mediante la adición de ruido gaussiano de valor medio 0 y varianza 0.01.
2. Implemente el algoritmo de filtrado adaptativo de reducción local del ruido, y aplíquelo a la imagen empleando los parámetros de ruido introducidos en el ítem anterior.
3. Aplique a la imagen degradada el filtro de la media geométrica y compare visualmente las imágenes de salida de ambos métodos.
4. Repita el ejercicio para valores de varianza mayores y analice que sucede.

Ejercicio 5: Eliminación de ruido periódico

1. La imagen `'img_degradada.tif'` está degradada por una interferencia sinusoidal. Muestre la imagen y su espectro de Fourier, y analice la información del ruido.
2. Localice los picos fundamentales del ruido. (semi-automático con clicks en la imagen o automático localizando las magnitudes de los picos del ruido).

3. Utilice estos parámetros para implementar los siguientes filtros:
   - un filtro rechazabanda ideal,
   - un filtro rechazabanda Gaussiano o de Butterworth,
   - un filtro notch ideal que elimine las zonas centrales de las frecuencias del ruido (y su conjugado).
   - un filtro notch Gaussiano o de Butterworth.

4. Aplique los filtros y compare cualitativamente las imágenes filtradas respecto de la imagen original `'img.tif'` y cuantitativamente mediante el cálculo del ECM. Ajuste los parámetros de los filtros para obtener mejores resultados.
   ¿Se anima a comparar los resultados con los de un pasa-bajos frecuencial implementado en la guía previa?

5. Obtenga la imagen de sólo ruido mediante la aplicación de un filtro pasabanda o un filtro notch pasante, de su elección.

6. Aplique los filtros y analice los resultados con las imágenes `'noisy_moon.jpg'` y `'HeadCT_degradada.tif'`.

Ejercicio 6: Restauración por filtrado de Wiener

1. Estudie el funcionamiento y los resultados de la aplicación del filtro de Wiener en las siguientes situaciones:
   (a) restauración de imágenes fuera de foco:
   `https://docs.opencv.org/trunk/de/d3c/tutorial_out_of_focus_deblur_filter.html`
   (b) restauración de imágenes con desenfoque por movimiento:
   `https://docs.opencv.org/trunk/d1/dfd/tutorial_motion_deblur_filter.html`

2. Implemente estos algoritmos en `Python`, migrando los códigos en `C++` disponibles en estas URLs.

3. Busque en internet una imagen con cada tipo de degradación y restaurelas. Analice y comente los resultados.

Ejercicio 7: Aplicación

Para las imágenes `FAMILIA_a.jpg`, `FAMILIA_b.jpg` y `FAMILIA_c.jpg`, identifique el tipo de ruido que afecta a cada una y calcule los parámetros estadísticos para dichos ruidos. Elija apropiadamente el mejor filtro para cada caso, ajuste los parámetros y restaure las imágenes.

Lecturas adicionales:

- BilateralFilter() y adaptiveBilateralFilter().
- FastNlMeansDenoising() y FastNlMeansDenoisingColored()
  `https://docs.opencv.org/3.4/d5/d69/tutorial_py_non_local_means.html` y en el pdf `'Non-Local Means Denoising.pdf'`