

Mejora de imagen: operaciones elementales

Miguel Jara Arroyo

10 de diciembre de 2024

Resumen

Este trabajo aborda el desafío de mejorar imágenes capturadas en condiciones de baja iluminación mediante técnicas de procesamiento que incluyen ajustes de intensidad, análisis de histogramas y operadores aritméticos. Para ello, se seleccionaron cuatro imágenes representativas de la base de datos *The Dark Face*[1], y se implementaron dos métodos principales de ajuste de iluminación, cada uno con sus respectivas ventajas e inconvenientes. Este estudio no solo destaca las fortalezas y limitaciones de cada técnica, sino que también plantea oportunidades para futuras investigaciones orientadas a optimizar el procesamiento de imágenes en aplicaciones prácticas y avanzadas.

1. Introducción

El objetivo principal de esta actividad es familiarizarse con las técnicas y herramientas comúnmente utilizadas para mejorar imágenes. En este caso, se emplearán métodos específicamente diseñados para optimizar la iluminación de cuatro imágenes seleccionadas de la base de datos *The Dark Face*, que contiene 6000 imágenes reales capturadas en condiciones de baja iluminación nocturna en diversos entornos, como edificios educativos, calles y parques [1].

2. Material y Métodos

2.1. Software y Hardware utilizados

El laboratorio se llevó a cabo en un ordenador personal con las siguientes características de *hardware*:

- Procesador: AMD Ryzen 7 2700X Eight-Core Processor.
- Memoria RAM: 16 GB.

En cuanto al *software* utilizado, se emplearon las siguientes herramientas:

- Entorno de desarrollo: *Anaconda* con *Jupyter Lab* [2], [3].
- Librerías de Python:
 - *numpy* para operaciones matemáticas [4].
 - *matplotlib.pyplot* para la generación de gráficos y visualización de datos [5].
 - *cv2 (OpenCV)* para el procesamiento de imágenes [6].
 - *Pillow (PIL)* para la manipulación de imágenes [7].
 - *requests* para la descarga de recursos en línea, si fuera necesario [8].

2.2. Datos

De entre todas las imágenes del conjunto *The Dark Face*, se han seleccionado las imágenes 8, 9, 1150 y 5884 que se pueden ver en la figura 1.

Se han seleccionado estas imágenes, buscando representar los distintos contextos que aparecen en el conjunto de datos original siendo la imagen 8 una imagen de una acera con un edificio con carteles de letras iluminadas, la imagen 9 representa una calle peatonal con un edificio de piedra, la imagen 1150 representa a un partido de baloncesto nocturno y la imagen 5884 representa una avenida ajetreada con una gran cantidad de tráfico y especial mención a la moto del centro que se encuentra en mucha oscuridad.

Al tener una muestra tan variada se consigue una mayor representatividad del conjunto de datos original.

2.3. Metodología

En esta sección se detalla el procedimiento seguido para desarrollar la actividad. El código empleado para implementar los métodos descritos está disponible en el *notebook* indicado en las referencias [9].

El primer paso consistió en analizar las imágenes para comprender sus características principales. Todas las imágenes tienen dimensiones de $720 \times 1080 \times 3$, y su distribución de intensidad fue evaluada mediante histogramas. En la figura 2 se presentan los histogramas correspondientes a los tres canales de color RGB. Estos muestran que, para la mayoría de

Imagen: 8.png



Imagen: 9.png



Imagen: 1150.png



Imagen: 5884.png

Figura 1: Cuatro imágenes seleccionadas del conjunto *The Dark Face*.

las imágenes, los valores de intensidad están comprendidos entre 0 y 50. Sin embargo, en la imagen del partido de baloncesto, que fue capturada en un entorno con luz artificial, los valores de intensidad se distribuyen en un rango mayor, de 0 a 100. Esto refleja la diferencia en las condiciones de iluminación.

El ajuste inicial de la iluminación se realizó mediante la siguiente operación, implementada utilizando el método `clip()` de la biblioteca NumPy:

$$I' = \text{clip}(I \cdot f, 0, 255)$$

donde I representa los valores de intensidad de los píxeles de la imagen original, f es el factor de ajuste, e I' es la imagen resultante tras la operación. La imagen procesada con un factor de $f = 2,5$ se muestra en la figura 3.

Aunque esta operación mejora las áreas oscuras, también genera un efecto de "quemado" en las regiones previamente bien iluminadas, lo que provoca la pérdida de información en estas áreas.

Para mitigar este problema, y teniendo en cuenta los resultados obtenidos en los histogramas de la figura 2, se propuso calcular dinámicamente el factor de iluminación,

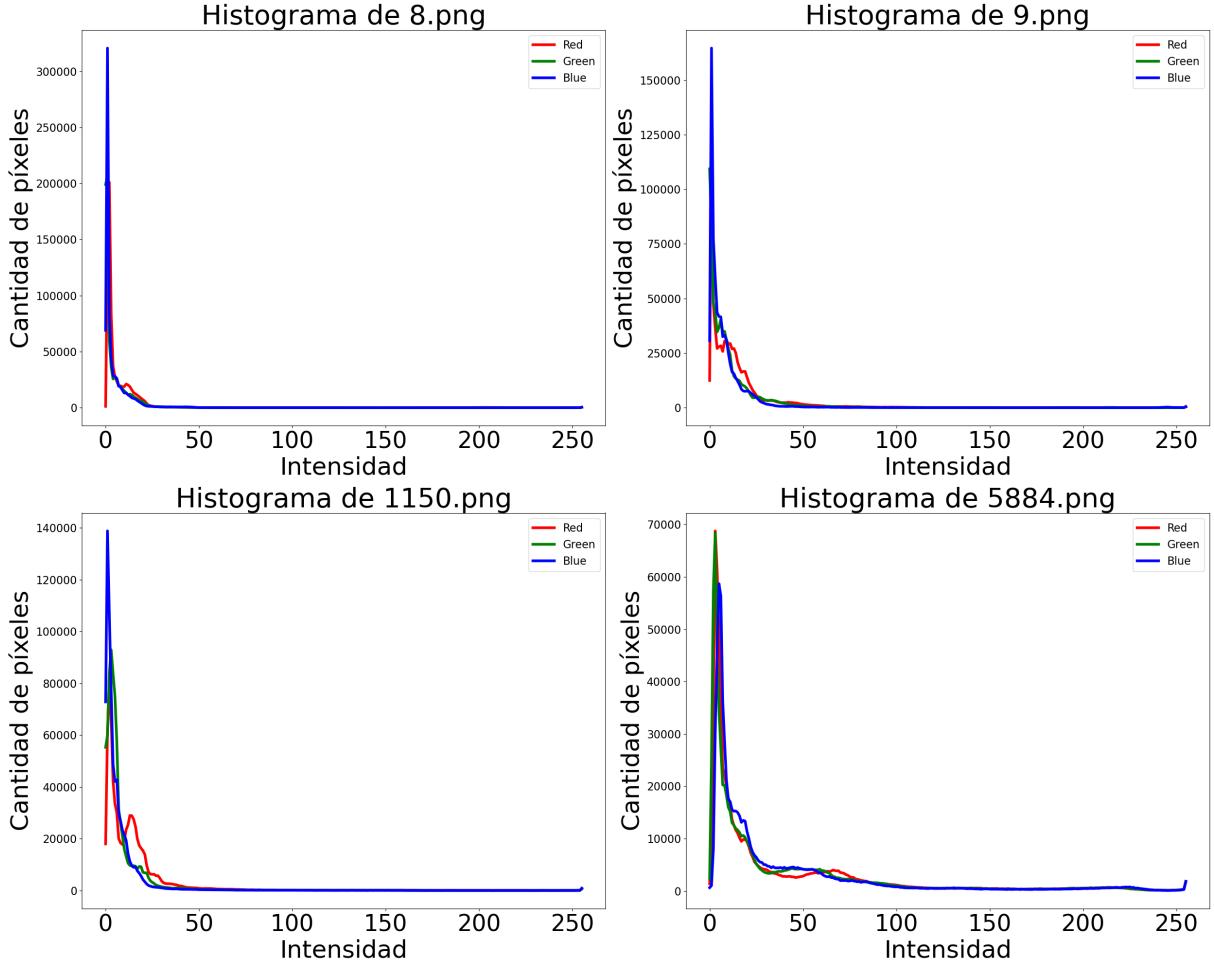


Figura 2: Histogramas de intensidad para los canales RGB de las imágenes analizadas.

adaptándolo a las características de cada imagen. Esto permite evitar el brillo excesivo en imágenes originalmente mejor iluminadas, como las del partido de baloncesto. Sin embargo, este enfoque introduce una limitación: la incapacidad de controlar explícitamente el nivel deseado de iluminación. Para superar esta restricción, se introdujo una variable denominada *iluminación_objetivo*, utilizada para ajustar el brillo de cada imagen hacia un valor predeterminado.

Para calcular el factor de ajuste dinámicamente, se propusieron dos aproximaciones:

- **Cálculo del factor dinámico por canal:** En esta aproximación, el ajuste se realiza de forma independiente para cada canal de la imagen (R, G y B). Para cada canal c , se calcula el factor como:

$$f_c = \frac{B_{\text{target}}}{\mu_c}, \quad \mu_c = \frac{1}{N_c} \sum_{i=1}^{N_c} I_{i,c}$$

donde B_{target} es la iluminación objetivo deseada, μ_c es el brillo promedio del canal



Figura 3: Imagen ajustada con un factor de iluminación de 2.5.

c , N_c es el número de píxeles en dicho canal, y $I_{i,c}$ es la intensidad del i -ésimo píxel en el canal c .

- **Cálculo del factor dinámico global:** En esta aproximación, se considera la imagen completa para el cálculo del ajuste. El factor se obtiene utilizando el brillo promedio de toda la imagen:

$$f = \frac{B_{\text{target}}}{\mu}, \quad \mu = \frac{1}{N} \sum_{i=1}^N I_i$$

Aquí, μ es el brillo promedio global de la imagen, N es el número total de píxeles, y I_i es la intensidad del i -ésimo píxel en la imagen.

Ambas aproximaciones ofrecen flexibilidad dependiendo de los requisitos específicos de la tarea. La primera permite un control más detallado en cada componente cromática, mientras que la segunda simplifica el cálculo al tratar la imagen como un todo.

3. Resultados

3.1. Calculo del factor con cada canal

En esta aproximación el cada canal se ilumina en función de su media de valores. Esto consigue una definición superior. Al ajustarse cada canal los mas finamente posible para ser iluminado, pero en contraposición las imágenes oscuras suelen potenciar mas el azul debido a que este suele ser mas escaso en estas imágenes, produciendo así una distorsión cromática. Tanto la mejora en la iluminación como la distorsión cromática se pueden ver



Figura 4: Resultado del ajuste de iluminación por canal sobre la cuatro imágenes.

en la figura 4, especialmente en la imagen "8.png" la cual es la mas oscura de las cuatro y por tanto ha quedado como la más azulada después del proceso de iluminación.

3.2. Calculo del factor global

En esta aproximación el cada canal se ilumina en función de la media de valores de la imagen. Esto consigue una definición un poco inferior al visto en los resultados obtenidos en el calculo del factor con cada canal, pero mantiene de una forma mejor el balance cromático, dando como resultado unas imágenes más naturales. Todo esto puede verse en la figura 5.

4. Conclusiones

En esta actividad se han explorado dos enfoques principales para ajustar la iluminación de imágenes, evaluando sus implicaciones tanto en la calidad visual como en la fidelidad cromática. Los resultados obtenidos destacan que cada método presenta ventajas y limitaciones específicas según el propósito del procesamiento.



Figura 5: Resultado del ajuste de iluminación global sobre las cuatro imágenes.

El ajuste por canal permite una manipulación precisa de los componentes cromáticos, lo cual mejora significativamente las áreas peor iluminadas de la imagen. Sin embargo, esta técnica tiende a introducir distorsiones cromáticas más notorias, comprometiendo la naturalidad de los colores. En contraste, el cálculo global del factor de ajuste ha demostrado mayor robustez en la preservación del equilibrio tonal, generando imágenes visualmente coherentes y con una representación más realista de los colores, aunque con una leve pérdida de detalle en las zonas más oscuras.

La elección entre ambos métodos depende del objetivo final del procesamiento. Para aplicaciones computacionales como la detección de rostros o la extracción de características, donde la cantidad de información procesada es crucial, el ajuste por canal resulta más adecuado. Sin embargo, si la prioridad es presentar imágenes a usuarios humanos, el método global ofrece una experiencia visual más natural y estéticamente agradable, posicionándolo como una opción superior en estos casos.

El análisis de estos enfoques sugiere que el desarrollo de modelos híbridos podría integrar las ventajas de ambas estrategias, permitiendo una adaptación dinámica según las características específicas de cada imagen. Asimismo, la implementación de algoritmos no

lineales se presenta como una vía prometedora para superar las limitaciones identificadas, logrando un procesamiento más eficiente y versátil.

Este trabajo contribuye al entendimiento y optimización de las técnicas de ajuste de iluminación en imágenes, estableciendo un marco teórico y práctico para futuras investigaciones orientadas a mejorar estas metodologías en escenarios más variados y exigentes.

Referencias

- [1] S. Rakshit, *Dark Face Dataset*, <https://www.kaggle.com/datasets/soumikrakshit/dark-face-dataset>, Último acceso: 10 de diciembre de 2024, 2020.
- [2] Anaconda, Inc., *Anaconda: The World's Most Popular Data Science Platform*, Disponible en: <https://www.anaconda.com/> (Accedido: 10 diciembre 2024), 2024.
- [3] Project Jupyter, *JupyterLab: A Web-Based Interactive Development Environment for Jupyter Notebooks*, Disponible en: <https://jupyter.org/> (Accedido: 10 diciembre 2024), 2024.
- [4] Harris, C. R., Millman, K. J. y v. et al., *NumPy: The fundamental package for array computing with Python*, Disponible en: <https://numpy.org/> (Accedido: 10 diciembre 2024), 2020.
- [5] Hunter, J. D., *Matplotlib: Visualization with Python*, Disponible en: <https://matplotlib.org/> (Accedido: 10 diciembre 2024), 2007.
- [6] Bradski, G. y Kaehler, A., *OpenCV: Open Source Computer Vision Library*, Disponible en: <https://opencv.org/> (Accedido: 10 diciembre 2024), 2000.
- [7] Clark, Alex et al., *Pillow: Python Imaging Library (PIL) Fork*, Disponible en: <https://python-pillow.org/> (Accedido: 10 diciembre 2024), 2024.
- [8] Kenneth Reitz, *Requests: HTTP for Humans*, Disponible en: <https://docs.python-requests.org/> (Accedido: 10 diciembre 2024), 2024.
- [9] A. del código o institución responsable, *Enlace al código utilizado*, Último acceso: 10 de diciembre de 2024, 2024.
Disponible en: https://drive.google.com/file/d/1fllZaZ003qjWWy3yG87dfsE_qodkmSHo/view?usp=sharing.