



## ****1. INTRODUCCIÓN****

Este informe tiene como objetivo aplicar técnicas de **procesamiento digital de imágenes** para resolver dos problemas principales:

1. **Ecualización local del histograma** de una imagen, con el fin de resaltar detalles ocultos que no pueden ser identificados mediante una ecualización global.
2. **Corrección automática de exámenes de tipo "multiple choice"** (opción múltiple), que incluye la validación del encabezado de cada examen y la generación de una imagen resumen indicando si los alumnos han aprobado o no.

El desarrollo de estos algoritmos se ha llevado a cabo en **Python**, utilizando las librerías **OpenCV**, **NumPy** y **Matplotlib**. En el contexto de cada problema, se explican los procedimientos implementados, los resultados obtenidos y el impacto de los parámetros utilizados en los resultados finales.

## ****2. PROBLEMA 1: Ecualización local del histograma****

### ****2.1. Descripción del problema****

La **ecualización del histograma** es una técnica de procesamiento de imágenes que permite mejorar el contraste de una imagen ajustando la distribución de los valores de intensidad de los píxeles. En este caso, se plantea extender la ecualización a un enfoque **local**, donde se define una ventana (vecindario) de tamaño MxN que se desplaza píxel por píxel sobre la imagen, y se aplica la ecualización del histograma dentro de cada ventana.

Este enfoque local es útil cuando se desea resaltar detalles en regiones específicas de la imagen que tienen características similares al fondo, las cuales podrían perderse al aplicar una ecualización global.

### ****2.2. Requisitos****

* Implementar una función que aplique la **ecualización local del histograma**.
* La función debe aceptar una imagen y los parámetros M y N que definen el tamaño de la ventana de procesamiento.
* El algoritmo debe recorrer la imagen píxel por píxel, utilizando el vecindario (ventana) centrado en cada píxel para aplicar la ecualización.

### ****2.3. Metodología****

Para resolver el problema, se siguieron los siguientes pasos:

1. **Análisis de la imagen**: Se realizó un análisis preliminar de la imagen para verificar su formato (escala de grises) y las características de los valores de intensidad de los píxeles.
2. **Expansión de la imagen**: Para poder aplicar la ecualización correctamente en los bordes de la imagen, se utiliza la función cv2.copyMakeBorder(), que expande la imagen replicando los bordes de la misma.
3. **Aplicación de la ecualización local**:
   * Se definió un doble bucle que recorre la imagen, centrando la ventana (M x N) en cada píxel.
   * Se extrae el vecindario de píxeles en la ventana, se aplana a una dimensión y se le aplica la ecualización con la función cv2.equalizeHist().
   * El valor del píxel central de la ventana se actualiza en la imagen resultante.
4. **Pruebas con diferentes tamaños de ventana**: Se analizaron diferentes tamaños de ventana (10x10, 20x20, 30x30 50x50) para observar cómo influyen en los resultados.

**2.4. Resultados**

* Con ventanas pequeñas (10x10), los detalles de zonas muy específicas de la imagen fueron resaltados, pero algunas áreas se vieron afectadas por el ruido.
* Con ventanas más grandes (30x30), se logró un mejor equilibrio entre resaltar detalles y mantener la homogeneidad de las áreas sin detalles.
* Ventanas demasiado grandes (50x50) comenzaron a perder los detalles locales, produciendo un efecto más cercano a la ecualización global, lo que es menos deseable en este caso.

La ventana de tamaño 50x50 resultó ser la más adecuada para resaltar los detalles sin generar ruido excesivo.

## ****3. PROBLEMA 2: Corrección automática de exámenes multiple choice****

### ****3.1. Descripción del problema****

En este segundo problema, se requiere desarrollar un sistema automático para la **corrección de exámenes de opción múltiple**. La entrada al sistema será una imagen del examen con respuestas y campos de encabezado (nombre, fecha y clase), y el sistema debe:

1. Detectar y validar las respuestas seleccionadas para cada pregunta.
2. Verificar si los campos del encabezado están correctamente completados.
3. Evaluar múltiples exámenes y generar una imagen resumen que indique los alumnos que aprobaron (con al menos 6 respuestas correctas) y los que no.

### ****3.2. Requisitos****

* El algoritmo debe detectar las respuestas seleccionadas en las 10 preguntas y compararlas con una clave de respuestas correcta.
* Validar los campos del encabezado con las siguientes restricciones:
  + **Name**: Al menos 2 palabras y no más de 25 caracteres.
  + **Date**: 8 caracteres en una sola palabra.
  + **Class**: Un único carácter.
* Para cada examen, se debe generar una salida indicando si cada respuesta es correcta o incorrecta.
* Generar una imagen de salida con los resultados de todos los exámenes, diferenciando visualmente los aprobados de los desaprobados.

### ****3.3. Metodología****

El enfoque para resolver este problema se divide en varias etapas:

#### ****3.3.1. Procesamiento de la imagen****

1. **Conversión a escala de grises y umbralización**:
   * Las imágenes de los exámenes se convierten a escala de grises, y luego se aplica una **umbralización** para obtener una imagen binaria donde los píxeles marcados se diferencien claramente de los no marcados.
2. **Detección de contornos**:
   * Utilizando cv2.findContours(), se detectan los contornos dentro de la imagen. Los contornos más grandes corresponden a las preguntas y opciones de respuesta.
3. **Recorte de preguntas**:
   * Se recortan las áreas correspondientes a las preguntas mediante un índice que asocia cada contorno con una pregunta específica.

#### ****3.3.2. Validación de las respuestas****

1. **Detección de la línea de respuesta**:
   * En cada pregunta, se busca el segundo contorno más grande (la línea de respuesta) y se recorta desde esta línea hacia arriba, capturando la opción seleccionada por el estudiante.
2. **Identificación de la letra seleccionada**:
   * A partir de las componentes conectadas, se identifica la letra marcada (A, B, C o D). Si no hay ninguna opción marcada o si hay más de una opción marcada, la respuesta se considera incorrecta.

#### ****3.3.3. Validación del encabezado****

El encabezado del examen se verifica según los siguientes criterios:

* **Nombre**: Se analiza la cantidad de palabras y la longitud total del nombre.
* **Fecha**: Se asegura que contenga exactamente 8 caracteres.
* **Clase**: Se verifica que sea un solo carácter.

#### ****3.3.4. Corrección del examen y generación de la imagen final****

1. **Corrección automática**:
   * Se compara cada respuesta detectada con la clave correcta:
     + Respuesta correcta = OK
     + Respuesta incorrecta = MAL
2. **Generación de la imagen resumen**:
   * Se recortan los campos "Name" de cada examen y se generan imágenes de salida diferenciando visualmente los alumnos que aprobaron (bordes verdes) de los que desaprobaron (bordes rojos).

### ****3.4. Resultados****

* Se implementó correctamente el sistema de corrección, validando tanto las respuestas como los encabezados de los exámenes.
* El algoritmo detectó y evaluó correctamente las opciones marcadas en los exámenes, y generó una imagen final con los nombres de los alumnos y su estado (aprobado/desaprobado).

## ****4. CONCLUSIONES****

El proyecto ha demostrado la eficacia de las técnicas de procesamiento de imágenes para resolver problemas tanto de análisis de imágenes como de automatización en tareas como la corrección de exámenes. Los resultados obtenidos en ambos problemas fueron satisfactorios:

1. En el caso de la **ecualización local del histograma**, se logró resaltar detalles no visibles. La elección del tamaño de la ventana es fundamental para obtener los mejores resultados.
2. Para la **corrección de exámenes multiple choice**, detecta respuestas, valida el encabezado y genera un informe visual de los resultados. Esto automatiza un proceso que típicamente se realiza de forma manual.