TRABAJO PRÁCTICO N° 1 - Año 2024 - 2° Semestre

Problema 1 - Ecualización local de histograma

La técnica de ecualización del histograma se puede extender para un análisis local, es decir, se puede realizar una ecualización local del histograma. El procedimiento sería definir una ventana cuadrada o rectangular (vecindario) y mover el centro de la ventana de pixel a pixel. En cada ubicación, se calcula el histograma de los puntos dentro de la ventana y se obtiene de esta manera, una **transformación local de ecualización del histograma**. Esta transformación se utiliza finalmente para mapear el nivel de intensidad del píxel centrado en la ventana bajo análisis, obteniendo así el valor del píxel correspondiente a la imagen procesada. Luego, se desplaza la ventana un píxel hacia el costado y se repite el procedimiento hasta recorrer toda la imagen.

Esta técnica resulta útil cuando existen diferentes zonas de una imagen que poseen detalles, los cuales se quiere resaltar, y los mismos poseen valores de intensidad muy parecidos al valor del fondo local de la misma. En estos casos, una ecualización global del histograma no daría buenos resultados, ya que se pierde la localidad del análisis al calcular el histograma utilizando todos los píxeles de la imagen.

Desarrolle una función en *python*, para implementar la ecualización local del histograma, que reciba como parámetros de entrada la imagen a procesar, y el tamaño de la ventana de procesamiento (M x N). Utilice dicha función para analizar la imagen que se muestra en la *figura 1* e informe cuales son los detalles escondidos en las diferentes zonas de la misma. Analice la influencia del tamaño de la ventana en los resultados obtenidos.

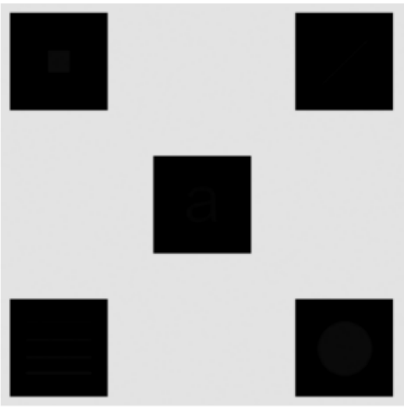
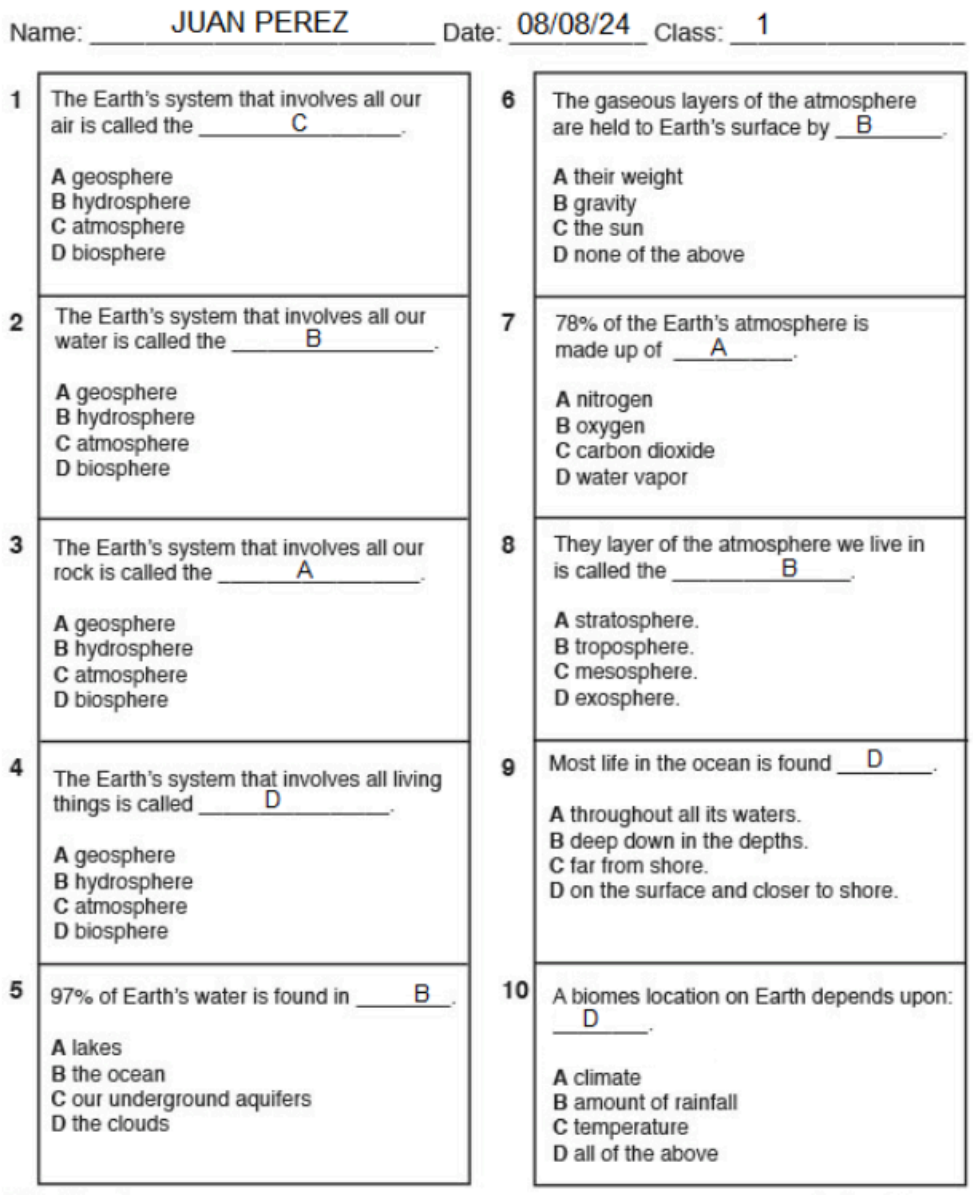


Figura 1: Imagen con detalles en diferentes zonas.

AYUDA: Con la siguiente función, puede agregar una cantidad fija de pixels a una imagen: ***cv2.copyMakeBorder(img, top, bottom, left, right, borderType)***, donde *top*, *bottom*, *left* y *right* son valores enteros que definen la cantidad de píxeles a agregar arriba, abajo, a la izquierda y a la derecha, respectivamente, y *borderType* define el valor a utilizar. Por ejemplo, ***borderType = cv2.BORDER_REPLICATE*** replica el valor de los bordes.

Problema 2 - Corrección de múltiple choice

En la *figura 2* se muestra el esquema de un examen múltiple choice de 10 preguntas con cuatro opciones para cada una de ellas (A, B, C, D). La plantilla también tiene un encabezado con tres campos para completar datos personales (*Name*, *Date* y *Class*).


Name: JUAN PEREZ Date: 08/08/24 Class: 1

1 The Earth's system that involves all our air is called the C.
A geosphere
B hydrosphere
C atmosphere
D biosphere

2 The Earth's system that involves all our water is called the B.
A geosphere
B hydrosphere
C atmosphere
D biosphere

3 The Earth's system that involves all our rock is called the A.
A geosphere
B hydrosphere
C atmosphere
D biosphere

4 The Earth's system that involves all living things is called D.
A geosphere
B hydrosphere
C atmosphere
D biosphere

5 97% of Earth's water is found in B.
A lakes
B the ocean
C our underground aquifers
D the clouds

6 The gaseous layers of the atmosphere are held to Earth's surface by B.
A their weight
B gravity
C the sun
D none of the above

7 78% of the Earth's atmosphere is made up of A.
A nitrogen
B oxygen
C carbon dioxide
D water vapor

8 They layer of the atmosphere we live in is called the B.
A stratosphere.
B troposphere.
C mesosphere.
D exosphere.

9 Most life in the ocean is found D.
A throughout all its waters.
B deep down in the depths.
C far from shore.
D on the surface and closer to shore.

10 A biomes location on Earth depends upon: D.
A climate
B amount of rainfall
C temperature
D all of the above

Figura 2: Esquema del exámen.

Se tiene una serie de exámenes resueltos, en formato de imagen, y se pretende corregirlos de forma automática por medio de un *script* en *python*. Para esto, asuma que las respuestas correctas son las siguientes:

1. C **2.** B **3.** A **4.** D **5.** B **6.** B **7.** A **8.** B **9.** D **10**. D

En el caso que alguna respuesta tenga marcada más de una opción, la misma se considera como incorrecta, de igual manera si no hay ninguna opción marcada.

El algoritmo a desarrollar debe resolver los siguientes puntos:

a. Debe tomar únicamente como entrada la imagen de un examen (**no usar como dato las coordenadas de las preguntas**) y mostrar por pantalla cuáles de las respuestas son correctas y cuáles incorrectas. Por ejemplo:

Pregunta 1: OK
Pregunta 2: MAL
Pregunta 3: OK
…
Pregunta 10: OK

b. Con la misma imagen de entrada, validar los datos del encabezado y mostrar por pantalla el estado de cada campo teniendo en cuentas las siguientes restricciones:

- **i.** *Name*: debe contener al menos dos palabras y no más de 25 caracteres.
- **ii.** *Date*: deben ser 8 caracteres formando una sola palabra.
- **iii.** *Class*: un único caracter.

Por ejemplo:

Name: OK
Date: MAL
Class: MAL

Asuma que todos los campos ocupan un solo renglón y que se utilizan caracteres alfanuméricos y barra inclinada " / ".

En la *figura 3a* se muestra un ejemplo donde los campos del formulario están todos correctamente cargados, mientras que en la *figura 3b* se muestra otro ejemplo donde todos los campos están cargados de forma incorrecta.

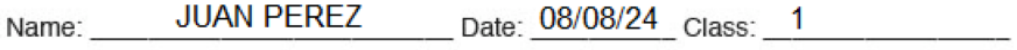


Figura 3a: Todos los campos cargados correctamente.

Figura 3b: Todos los campos cargados incorrectamente.

c. Utilice el algoritmo desarrollado para evaluar las imágenes de exámenes resueltos (archivos *examen_<id>.png*) e informe cada resultado obtenido.

d. Generar una imagen de salida informando los alumnos que han aprobado el examen (con al menos 6 respuestas correctas) y aquellos alumnos que no. Esta imagen de salida debe tener los “crop” de los campos *Name* del encabezado de todos los exámenes del punto anterior y diferenciar de alguna manera aquellos que corresponden a un examen aprobado de uno desaprobado.

AYUDA:

- Existen varias formas de detectar las celdas de cada pregunta, una de ellas es detectando las coordenadas de las líneas verticales y horizontales que alinean dichas celdas. Para ello, una opción es primero umbralizar la imagen ***img_th=img<th*** y luego sumar el valor de los píxeles que están en cada columna para detectar las líneas verticales ***img_cols=np.sum(img_th_ones,0)*** y sumar el valor de los píxeles que están en cada fila para detectar las líneas horizontales ***img_rows=np.sum(img_th_ones,1)***. Luego, dado que en dichas líneas existen muchos más píxeles que en las demás partes del formulario, se puede definir un umbral acorde (uno para las líneas horizontales y otro para las líneas verticales) y detectar así las posiciones de las mismas con ***img_rows_th=mg_rows>th_row***. Tenga en cuenta que las líneas pueden tener más de un píxel de ancho, por lo cual, quizás deba encontrar el principio y el fin de las mismas en la variable ***img_rows_th***. Esta misma técnica se puede utilizar para detectar las líneas del encabezado y obtener *subimagenes* de los campos del mismo.

- Con las *subimagenes* de las celdas detectadas, una posible forma de obtener las respuestas en cada una de ellas es obteniendo las componentes conectadas dentro: ***cv2.connectedComponentsWithStats(celda_img,8,cv2.CV_32S)***. Tenga especial cuidado que no hayan quedado píxeles de las líneas divisorias de la tabla dentro de la celda. Una posible forma de evitar este problema, es eliminar las componentes conectadas de área muy chica, definiendo un umbral: ***ix_area=stats[:,-1]>th_area*** y luego ***stats=stats[ix_area,:]***.

- Para diferenciar entre las cuatro posibles letras de las respuestas (A, B, C y D) puede obtener los contornos de cada una de ellas y analizar sus formas, tamaños, cantidad de contornos hijos, entre otras características.