**INTELIGENCIA ARTIFICIAL – ST414**

**EXAMEN FINAL**

Desarrollo grupal, se entrega el martes 9 de julio.

**5 de julio de 2019**

# OBJETIVO.

Diseñar un procedimiento para romper captchas usando técnicas de image matching, con esta finalidad el grupo deberá descargar un conjunto de imágenes desde la página del MEF SIAF usando el paquete selenium, deberá de preparar las imágenes, deberá etiquetar las imágenes manualmente y luego deberá de crear un procedimiento para encontrar las regiones en la imagen donde existe mayor similaridad.

# CASO

Los grupos de trabajo diseñan un procedimiento para identificar las letras presentadas en la imagen descargada.

Con este fin debe de ejecutar como mínimo los siguientes pasos:

## PROCESO DE APRENDIZAJE

1. Descargar al menos 1000 imágenes de la página de MEF SIAF, mediante el procedimiento que use el paquete selenium (ver archivo adjunto).
2. Cada imagen descargada debe ser etiquetada con su contenido y un número de secuencia (prhqw\_0001), para evitar su confusión.

|  |  |
| --- | --- |
| Imagen | Nombre |
|  | 5hgnr\_0001 |
|  | ceg58\_0001 |
|  | wnb2g\_0001 |
|  | pkkr2\_0001 |
|  | pkkr2\_0002 |

1. Binarizar la imagen, no es necesario segmentar las imágenes.

|  |  |
| --- | --- |
|  |  |

En caso de que sea necesario, es recomendable que parta la imagen en 2, luego binarice y luego los vuelva a juntar.

1. Invierta el color de las imágenes.

|  |  |
| --- | --- |
|  |  |

1. Segmente las imágenes, en sus letras constituyentes (puede ser manualmente).

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |

1. Prepare la plantilla de cada carácter (puede ser manualmente).

Segmente las imágenes

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |

La plantilla de un carácter (o de cualquier objeto) corresponde al segmento de la imagen original que solo contiene el carácter, abarcando la forma de la imagen, usted debe de hacerlo manualmente.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |

Prepare una plantilla por cada uno de los caracteres a reconocer.

## PROCESO DE CONSULTA.

1. Ejecute los pasos 1 al 5 de la etapa anterior, observe que la región de interés corresponde a la sección “blanca”
2. Considerando una solo imagen, aplicar el siguiente procedimiento:

* “Convolucionar la imagen obtenida en el paso 7, con cada una de las plantillas preparadas en el paso 6, Use la función filter2D del opencv.

SE CONVOLUCIONA CADA LETRA DE LAS PLANTILLA , TODAS LAS A , TODAS LAS B…

|  |
| --- |
| weightDm = cv2.filter2D(imCaptcha, -1, np.array(template)) |

1. En la imagen convolucionada identificar la región de mayor ranking.

* Identificar el carácter que tenga el mayor ranking.
* Tenga cuidado en este paso dado que puede ser que los tamaños de los templates sean de diferentes tamaños, aplique algún procedimiento de normalización para que sean comparables.
* Por otro lado, no basta con que sea el mayor, dado que pueden existir varios mayores o puede ser que la diferencia entre ellos sea muy pequeña.

## EVALUACIÓN DEL RENDIMIENTO

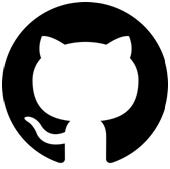
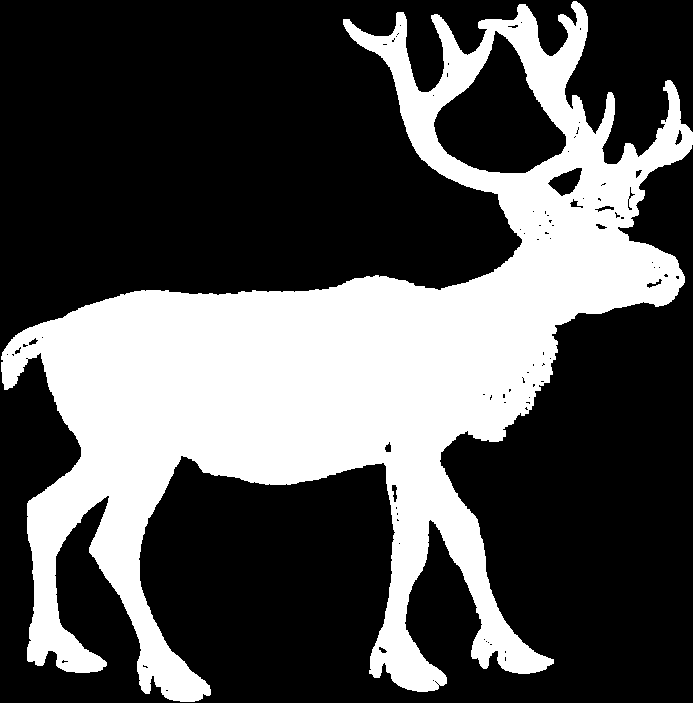
Preparar un procedimiento para medir lo siguiente:

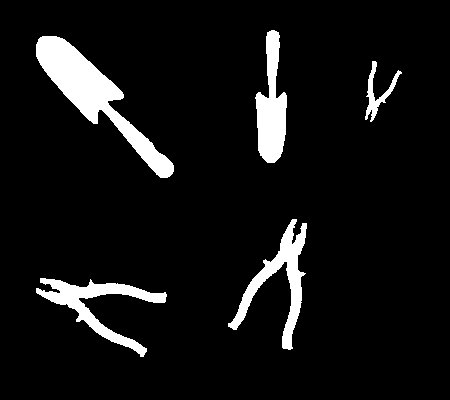
1. Ranking de éxito por cada carácter. Prepare una lista indicando de todas las letras a, cuantos fueron reconocidas, de todas las b cuantos fueron reconocidos y así sucesivamente. Entregue la lista rankeada.
2. Ranking de éxito por imagen. Diga de todas las imágenes trabajadas en cuantos de ellas se ha reconocido al menos 3 caracteres.
3. Ranking de éxito por imagen completa. Diga de todas las imágenes trabajadas en cuantos de ellas se ha reconocido todos los caracteres simultaneamente.

# BÚSQUEDA DE PATRONES MORFOLÓGICOS EN IMÁGENES BINARIAS.

## Imagen binaria

Una imagen binaria es una imagen digital que tiene únicamente dos valores posibles para cada píxel, estos son equivalentes a negro y blanco (aunque puede usarse cualquier pareja de colores). Uno de los colores se emplea como fondo y el otro para los objetos que aparecen en la imagen. Las imágenes binarias se utilizan para gestionar **patrones morfológicos**, sin interesar el nivel de gris o el color.



## Patrón morfológico

En imágenes binarias, un **patrón morfológico** es un conjunto de píxeles dispuestos de tal forma que tienen un significado para un humano, que puede ser identificable.

El **patrón morfológico** se puede caracterizar mediante una **imagen patrón** o template.

|  |  |
| --- | --- |
|  |  |
| **Imagen Patrón** | **Imagen de Entrada** |

Estos **patrones morfológicos** si bien son "fáciles" de identificar para una persona, no son fáciles de identificar para un computador dado la cantidad de operaciones aritméticas que debe de realizar.

## Imagen Patrón (Plantilla o Template)

Una **imagen patrón** es una plantilla es una imagen que contiene el patrón de pixeles que se desea identificar.

Dado la naturaleza de las imágenes, no es sencillo definir una plantilla para un computador, dado que el patrón puede:

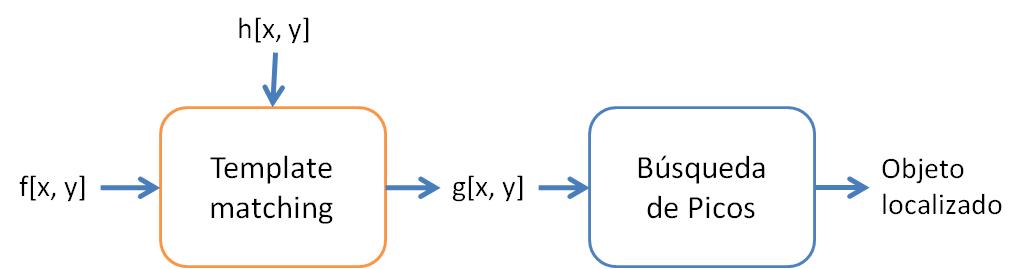
1. Estar rotado
2. Tener diferentes tamaños
3. Contener ruido.

## Búsqueda de patrones morfológicos

La búsqueda de patrones es una técnica de análisis de imágenes que se utiliza para buscar una **imagen patrón** o plantilla, dentro de una imagen de mayor tamaño, no se busca apariciones exactas o iguales al patrón, se busca regiones que permitan un grado de variación respecto al patrón original.

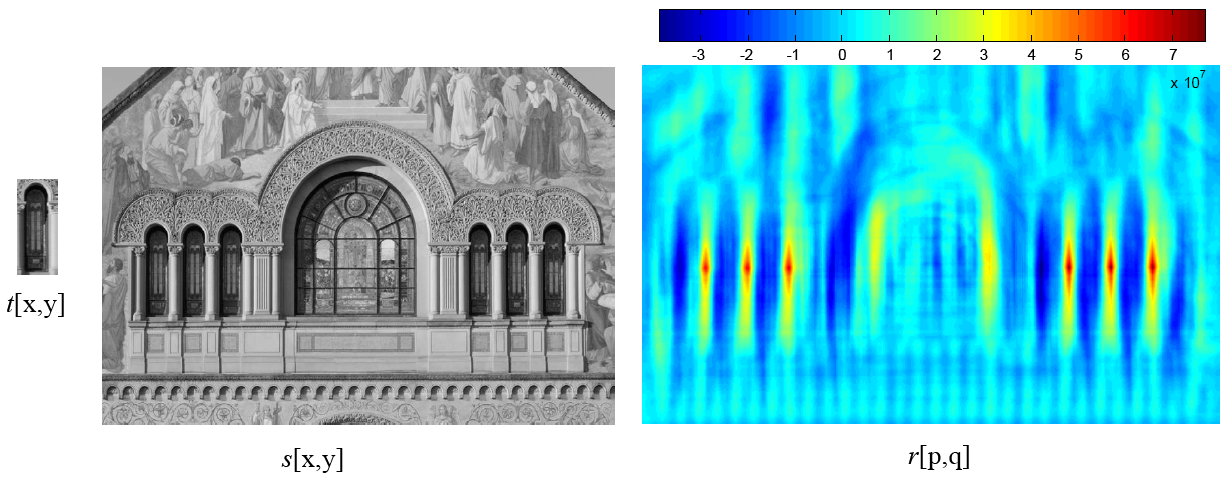
La técnica tiene diferentes denominaciones:

1. Template matching
2. Búsqueda de patrones de morfológicos
3. Búsqueda de patrones de imágenes
4. Comparación de plantillas



El procedimiento usa una **imagen patrón** (h[x, y]), que se busca en la imagen de entrada (f[x,y]) para producir un **ranking de similitud** (g[x, y]).

El siguiente procedimiento busca los picos en el ranking de similitud para encontrar el centroide de la región de interés (o de las regiones de interés).



f[x, y]

h[x, y]

g[x, y]

## Convolución

Una convolución es una operación matemática que transforma dos matrices (f) y (h) en una nueva matriz (g). A la matriz h se le denominará "kernel", "filtro" o "mascara".

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| -Matriz imagen (f)   |  |  |  |  | | --- | --- | --- | --- | | 15 | 20 | 101 | 100 | | 200 | 50 | 55 | 8 | | 10 | 11 | 230 | 202 | | 100 | 130 | 115 | 120 | | -Máscara (h)   |  |  |  | | --- | --- | --- | | 1 | 0 | 1 | | 0 | 1 | 0 | | 1 | 0 | 1 | |

Los pasos para realizar la convolución son los siguientes:

1. **Sobreponer el centro de la máscara (kernel) sobre un elemento de la matriz f.**

[15 **1** 20 **0** 101 **1** 100]

[200 **0** 50 **1** 55 **0** 8]

[10 **1** 11 **0** 230 **0** 202]

[100 130 115 120]

1. **Multiplicar cada valor (peso) de la máscara por el píxel de la matriz de imagen que se encuentra "bajo" la máscara.**

El procedimiento lee sucesivamente la matriz f, de izquierda a derecha y de arriba a abajo, aplicando el kernel a todos los píxeles del área de acción del kernel.

Se multiplica el valor de cada uno de ellos por el valor correspondiente del kernel y se suman los resultados:

(15\*1), (20\*0), (101\*1), (200\*0), (50\*1), (55\*0), (10\*1), (11\*0), (230\*0).

1. **Sumar los productos individuales en el apartado anterior.**

(15\*1) + (20\*0) + (101\*1) + (200\*0) + (50\*1) + (55\*0) + (10\*1) + (11\*0) + (230\*0).

Obteniendo 406

El resultado final es:

|  |  |  |  |
| --- | --- | --- | --- |
| 65 | 166 | 151 | 101 |
| 75 | 406 | 381 | 331 |
| 60 | 290 | 280 | 230 |
| 10 | 240 | 230 | 230 |

Observar que la matriz resultando (g) es del mismo tamaño que de la imagen de entrada (f), en el caso de los bordes solo se aplica las regiones que se encuentran dentro de la matriz original, asumiendo que los valores fuera son cero (0).

# ALCANCES DEL TRABAJO

1. Recolección de los casos de prueba. El grupo define un conjunto de imágenes que serán usadas para diseñar el filtro. (2 puntos)
2. Diseño de la arquitectura de la solución. (6 puntos)
3. Construcción del procedimiento de solución (10 puntos)
   1. Diseño del procedimiento de “aprendizaje”
   2. Diseño del procedimiento de “consulta”
   3. Diseño del procedimiento de evaluación.
4. Pruebas del procedimiento. (2 puntos)
5. Redacción de informe técnico en español. (0 puntos)
6. Código fuente de los programas desarrollados (0 puntos)

# RECOMENDACIONES

* Recuerde que las imágenes no hablan por si mismas, es necesario explicarlas y si es con ejemplos mejor.
* Numere todas sus tablas, figuras, fórmulas y otros elementos del documento.
* Recuerde que el modelo de solución es el plano del software, por lo que es necesario que exista correspondencia entre el modelo de solución y si programa fuente.
* Entregue el código fuente de tal forma que sea legible por el lector de su trabajo, evite solo pegar el código y entregarlo, asígnele un número y referéncielo desde su explicación.
* Explique los procedimientos desarrollados, presentando ejemplos de las imágenes procesadas.

# LENGUAJE DE PROGRAMACIÓN

Para el desarrollo del proyecto se considera el Python.

# MOTIVACIÓN

Informes preparados en latex, tienen **5 puntos adicionales** en la nota de ese examen, hasta llegar a 20 puntos.

# PENALIDADES

Los trabajos donde no se observe congruencia entre el modelo de solución (arquitectura del software) y el código fuente entregado, tendrá una **reducción de 6 puntos** en la nota final.

La congruencia implica que debe de existir correspondencia entre ellos.

|  |  |
| --- | --- |
|  | def main\_processes(pA, pB):  tempP = pA + pB;  return tempP  if \_\_name\_\_ == '\_\_main\_\_':  A = 1  B = 2  P = main\_processes(A, B)  print(P) |
|  | def main\_processes(pA, pB):  tempS = F1(pA)  tempQ = F2(pB)  tempP = F3(tempS, tempQ)  return tempP  def F1(pA):  tempP = pA  return tempP  def F2(pB):  tempP = pB  return tempP  def F3(pS, pQ):  tempP = pS + pQ  return tempP  if \_\_name\_\_ == '\_\_main\_\_':  A = 1  B = 2  P = main\_processes(A, B)  print(P) |