

# PRÁCTICA: Unidad 2 - Transformación y filtrado.

# Ejercicio 1

A partir de la imagen mostrada en la figura 1 (archivo *faces.jpg*), se requiere aplicar un **efecto de borrosidad** a cada rostro presente en ella. Para lograr esto, será necesario procesar la imagen utilizando **técnicas de segmentación** adecuadas y, posteriormente, aplicar el filtro de borrosidad correspondiente a las áreas identificadas.

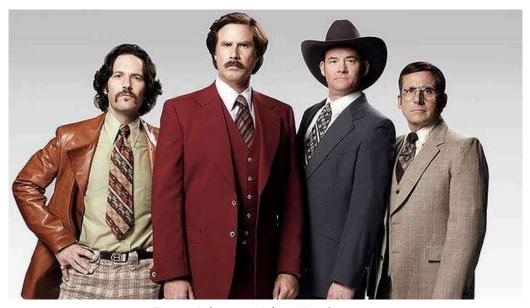


Figura 1: Imagen del archivo faces.jpg

# 1.1 Carga de la imagen de entrada.

- a) Cargar la imagen desde el archivo faces.jpg y mostrarla en una figura.
- b) Obtener y mostrar la información básica de la imagen (tipo de dato, dimensiones, valores máximos y mínimos de intensidad en cada canal).

#### 1.2 Filtrado manual.

- c) Ubicar cada rostro de la imagen, encerrarlos en un rectángulo (bounding box) y mostrarlos en una nueva figura (obtener las coordenadas de los rostros de forma manual).
- d) Tomando como base la imagen original, recortar y mostrar los rostros en una única figura utilizando *subplots* para cada rostro.

UNR | TUIA | PDI



# Universidad Nacional de Rosario Facultad de Ciencias Exactas, Ingeniería y Agrimensura Tecnicatura Universitaria en Inteligencia Artificial Procesamiento de Imágenes I - IA 4.4

- e) Filtrar cada recorte con el método *cv2.blur()*, utilizando los parámetros adecuados para obtener el efecto de borrosidad deseado, y mostrar cada resultado en una única figura con el uso de *subplots* para cada rostro.
- f) Pegar cada recorte con efecto de borrosidad en las posiciones correspondientes de la imagen original y mostrar el resultado en una nueva figura.

#### 1.3 Filtrado automático.

- g) A partir de la imagen original, convertir la misma a escala de grises con el método cv2.cvtColor(), mostrarla en una figura y obtener su información básica (tipo de dato, dimensiones, valores máximos y mínimos de intensidad).
- h) Ubicar cada rostro mediante el uso de un modelo pre-entrenado de detección de rostros basado en un clasificador en *Haar Cascade* (ver documentación):
  - i) Crear el objeto clasificador con cv2. Cascade Classifier().
  - ii) Cargar el modelo *cv2.data.haarcascades* + *'haarcascade\_frontalface\_alt.xml'* con el método *load()* del objeto creado.
  - iii) Aplicar el clasificador sobre la imagen en escala de grises con el método detectMultiScale() del objeto creado.
- i) Encerrar cada rostro detectado en un rectángulo (bounding box), sobre la imagen original, y mostrarlos en una nueva figura.
- j) Tomando como base la imagen original, recortar y mostrar los rostros detectados en una única figura utilizando *subplots* para cada rostro.
- k) Filtrar cada recorte con el método *cv2.GaussianBlur()*, utilizando los parámetros adecuados para obtener el efecto de borrosidad deseado y mostrar cada resultado en una única figura con el uso de *subplots* para cada rostro.
- I) Pegar cada recorte con efecto de borrosidad en las posiciones correspondientes de la imagen original y mostrar el resultado en una nueva figura.

UNR | TUIA | PDI



Universidad Nacional de Rosario Facultad de Ciencias Exactas, Ingeniería y Agrimensura Tecnicatura Universitaria en Inteligencia Artificial Procesamiento de Imágenes I - IA 4.4

### Ejercicio 2

La figura 2 (archivo *john\_canny\_bio.png*) corresponde a una biografía de John F. Canny, en este ejercicio se requiere que, a partir de las técnicas de PDI vistas en clase, se recupere de forma automática el dato perdido debajo de la "mancha" que se observa en el pie de página de la misma. Tener en cuenta que es necesario no perder detalles ni información de la biografía en el proceso de recuperación. Una vez obtenido el dato perdido, se pide insertarlo en la imagen original y mostrar el resultado en una nueva figura.

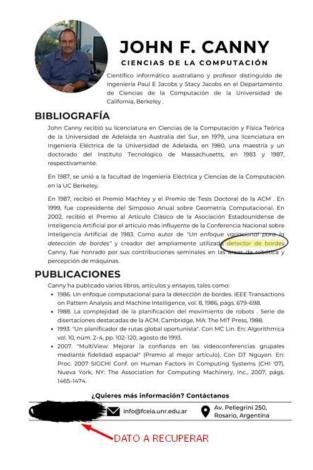


Figura 2: Imagen del archivo john\_canny\_bio.png

UNR | TUIA | PDI