

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.

- Cargar la imagen desde el archivo *faces.jpg* y mostrarla en una figura.
- 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.

- 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).
- Tomando como base la imagen original, recortar y mostrar los rostros en una única figura utilizando *subplots* para cada rostro.




- 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.CascadeClassifier()`.
 - ii) Cargar el modelo `cv2.data.harcascades + '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.
- l) Pegar cada recorte con efecto de borrosidad en las posiciones correspondientes de la imagen original y mostrar el resultado en una nueva figura.

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.



JOHN F. CANNY

CIENCIAS DE LA COMPUTACIÓN

Científico informático australiano y profesor distinguido de ingeniería Paul E Jacobs y Stacy Jacobs en el Departamento de Ciencias de la Computación de la Universidad de California, Berkeley.

BIBLIOGRAFÍA

John Canny recibió su licenciatura en Ciencias de la Computación y Física Teórica de la Universidad de Adelaida en Australia del Sur, en 1979, una licenciatura en Ingeniería Eléctrica de la Universidad de Adelaida, en 1980, una maestría y un doctorado del Instituto Tecnológico de Massachusetts, en 1983 y 1987, respectivamente.

En 1987, se unió a la facultad de Ingeniería Eléctrica y Ciencias de la Computación en la UC Berkeley.



En 1987, recibió el Premio Machtey y el Premio de Tesis Doctoral de la ACM. En 1999, fue copresidente del Simposio Anual sobre Geometría Computacional. En 2002, recibió el Premio al Artículo Clásico de la Asociación Estadounidense de Inteligencia Artificial por el artículo más influyente de la Conferencia Nacional sobre Inteligencia Artificial de 1983. Como autor de “Un enfoque computacional para la detección de bordes” y creador del ampliamente utilizado **detector de bordes**, Canny, fue honrado por sus contribuciones seminales en las áreas de robótica y percepción de máquinas.

PUBLICACIONES

Canny ha publicado varios libros, artículos y ensayos, tales como:

- 1986. Un enfoque computacional para la detección de bordes. IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 8, 1986, págs. 679-698.
- 1988. La complejidad de la planificación del movimiento de robots. Serie de disertaciones destacadas de la ACM, Cambridge, MA: The MIT Press, 1988.
- 1993. “Un planificador de rutas global oportunista”. Con MC Lin. En: Algorithmica vol. 10, núm. 2-4, pp. 102-120, agosto de 1993.
- 2007. “MultiView: Mejorar la confianza en las videoconferencias grupales mediante fidelidad espacial” (Premio al mejor artículo). Con DT Nguyen. En: Proc. 2007 SIGCHI Conf. on Human Factors in Computing Systems (CHI '07). Nueva York, NY: The Association for Computing Machinery, Inc., 2007, págs. 1465-1474.

¿Quieres más información? Contáctanos

 info@fceia.unr.edu.ar  Av. Pellegrini 250, Rosario, Argentina

DATO A RECUPERAR

Figura 2: Imagen del archivo *john_canny_bio.png*