Procesamiento de Información Visual

Tema 3. Segmentación y reconocimiento

José Manuel Fuertes García
Manuel J. Lucena López

Departamento de Informática
Universidad de Jaén

5 de abril de 2021

Índice

1. Segmentación
2. Paradigmas de segmentación
3. Clasificación y reconocimiento
4. Clasificación y reconocimiento mediante *deep learning*

Segmentación de imágenes

- La segmentación subdivide la imagen en una serie de regiones, que comparten alguna característica común:
  - Pertenecer al mismo objeto, tener el mismo color o textura, etc.

Segmentación de imágenes

- La segmentación subdivide la imagen en una serie de regiones, que comparten alguna característica común:
  - Pertenecer al mismo objeto, tener el mismo color o textura, etc.
- La segmentación de imágenes no triviales es una de las tareas más complejas en procesamiento de imágenes.

Segmentación de imágenes

- La segmentación subdivide la imagen en una serie de regiones, que comparten alguna característica común:
  - Pertenecer al mismo objeto, tener el mismo color o textura, etc.
- La segmentación de imágenes no triviales es una de las tareas más complejas en procesamiento de imágenes.

Segmentación de imágenes

- La segmentación subdivide la imagen en una serie de regiones, que comparten alguna característica común:
  - Pertenecer al mismo objeto, tener el mismo color o textura, etc.
- La segmentación de imágenes no triviales es una de las tareas más complejas en procesamiento de imágenes.

¿Cuáles son las regiones de interés?

Segmentación de imágenes

- La segmentación subdivide la imagen en una serie de regiones, que comparten alguna característica común:
  - Pertenecer al mismo objeto, tener el mismo color o textura, etc.
- La segmentación de imágenes no triviales es una de las tareas más complejas en procesamiento de imágenes.

¿Cuáles son las regiones de interés?
¿Cómo caracterizarlas?

Índice

1. Segmentación
2. Paradigmas de segmentación
3. Clasificación y reconocimiento
4. Clasificación y reconocimiento mediante *deep learning*

Discontinuidad y similaridad

- La mayoría de los algoritmos de segmentación se basan en uno de los dos principios.

Discontinuidad y similaridad

- La mayoría de los algoritmos de segmentación se basan en uno de los dos principios.

Discontinuidad

- Cambios bruscos en el valor de los píxeles (intensidad, color, etc.)

Discontinuidad y similaridad

- La mayoría de los algoritmos de segmentación se basan en uno de los dos principios.

Discontinuidad

- Cambios bruscos en el valor de los píxeles (intensidad, color, etc.)
- Detección de esos cambios

Discontinuidad y similaridad

- La mayoría de los algoritmos de segmentación se basan en uno de los dos principios.

Discontinuidad

- Cambios bruscos en el valor de los píxeles (intensidad, color, etc.)
- Detección de esos cambios $\longrightarrow$ fronteras.

Discontinuidad y similaridad

- La mayoría de los algoritmos de segmentación se basan en uno de los dos principios.

Discontinuidad

- Cambios bruscos en el valor de los píxeles (intensidad, color, etc.)
- Detección de esos cambios $\longrightarrow$ fronteras.

Similaridad

- Agrupaciones conexas de píxeles con valores parecidos

Discontinuidad y similaridad

- La mayoría de los algoritmos de segmentación se basan en uno de los dos principios.

Discontinuidad

- Cambios bruscos en el valor de los píxeles (intensidad, color, etc.)
- Detección de esos cambios $\longrightarrow$ fronteras.

Similaridad

- Agrupaciones conexas de píxeles con valores parecidos $\longrightarrow$ regiones.

El proceso de segmentación

Definición formal:

Siendo R el conjunto de píxeles de una imagen, y $Q(\cdot)$ un predicado aplicable a cualquier subconjunto de R, definiremos la segmentación de R como una serie de subconjuntos $R_i \subseteq R$, denominados *regiones*, que cumplen:

El proceso de segmentación

Definición formal:

Siendo R el conjunto de píxeles de una imagen, y $Q(\cdot)$ un predicado aplicable a cualquier subconjunto de R, definiremos la segmentación de R como una serie de subconjuntos $R_i \subseteq R$, denominados *regiones*, que cumplen:

a) $\bigcup_{i=1}^{n} R_i = R.$

El proceso de segmentación

Definición formal:

Siendo R el conjunto de píxeles de una imagen, y $Q(\cdot)$ un predicado aplicable a cualquier subconjunto de R, definiremos la segmentación de R como una serie de subconjuntos $R_i \subseteq R$, denominados *regiones*, que cumplen:

a) $\bigcup_{i=1}^{n} R_i = R.$

b) R_i es un conjunto conexo para todo i.

El proceso de segmentación

Definición formal:

Siendo R el conjunto de píxeles de una imagen, y $Q(\cdot)$ un predicado aplicable a cualquier subconjunto de R, definiremos la segmentación de R como una serie de subconjuntos $R_i \subseteq R$, denominados *regiones*, que cumplen:

a) $\bigcup_{i=1}^{n} R_i = R$.

b) R_i es un conjunto conexo para todo i.

c) $R_i \cap R_j = \emptyset$ para todo $i \neq j$.

El proceso de segmentación

Definición formal:

Siendo R el conjunto de píxeles de una imagen, y $Q(\cdot)$ un predicado aplicable a cualquier subconjunto de R, definiremos la segmentación de R como una serie de subconjuntos $R_i \subseteq R$, denominados *regiones*, que cumplen:

a) $\bigcup_{i=1}^{n} R_i = R$.

b) R_i es un conjunto conexo para todo i.

c) $R_i \cap R_j = \emptyset$ para todo $i \neq j$.

d) $Q(R_i) =$ Verdadero para todo R_i.

El proceso de segmentación

Definición formal:

Siendo R el conjunto de píxeles de una imagen, y $Q(\cdot)$ un predicado aplicable a cualquier subconjunto de R, definiremos la segmentación de R como una serie de subconjuntos $R_i \subseteq R$, denominados *regiones*, que cumplen:

a) $\bigcup_{i=1}^{n} R_i = R$.

b) R_i es un conjunto conexo para todo i.

c) $R_i \cap R_j = \emptyset$ para todo $i \neq j$.

d) $Q(R_i) =$ Verdadero para todo R_i.

e) $Q(R_i \cup R_j) =$ Falso para cualesquiera dos regiones R_i, R_j adyacentes.

Segmentación basada en fronteras

- En el caso ideal, la salida de un detector de fronteras delimitará las regiones de interés.

Segmentación basada en fronteras

- En el caso ideal, la salida de un detector de fronteras delimitará las regiones de interés.
- Debido al ruido, variaciones de iluminación y otras causas, las fronteras presentan discontinuidades.

Segmentación basada en fronteras

- En el caso ideal, la salida de un detector de fronteras delimitará las regiones de interés.
- Debido al ruido, variaciones de iluminación y otras causas, las fronteras presentan discontinuidades.
- Es necesario conectar los diferentes segmentos para poder delimitar las regiones de interés.

Segmentación basada en fronteras

- En el caso ideal, la salida de un detector de fronteras delimitará las regiones de interés.
- Debido al ruido, variaciones de iluminación y otras causas, las fronteras presentan discontinuidades.
- Es necesario conectar los diferentes segmentos para poder delimitar las regiones de interés.

- Estudiaremos varios tipos de técnicas, basadas en aproximaciones más o menos *locales* a las fronteras detectadas.

Procesamiento local

- Emplea información del vecindario más próximo a cada punto etiquetado como frontera por el detector.

Procesamiento local

- Emplea información del vecindario más próximo a cada punto etiquetado como frontera por el detector.
- Todos los puntos considerados *similares* según un criterio previo son enlazados.

Procesamiento local

- Emplea información del vecindario más próximo a cada punto etiquetado como frontera por el detector.
- Todos los puntos considerados *similares* según un criterio previo son enlazados.
- Normalmente, el criterio se construye a partir de la magnitud y la orientación del vector gradiente de cada punto.

Procesamiento local

- Emplea información del vecindario más próximo a cada punto etiquetado como frontera por el detector.
- Todos los puntos considerados *similares* según un criterio previo son enlazados.
- Normalmente, el criterio se construye a partir de la magnitud y la orientación del vector gradiente de cada punto.

- Dado un entorno S_{xy} de un píxel en (x, y), se considera que un punto $(s, t) \in S_{xy}$ es *similar* en magnitud al píxel (x, y) si:

$$|M(s,t) - M(x,y)| \leq \varepsilon$$

donde $M(\cdot)$ es la magnigud del vector gradiente, y ε es un umbral positivo.

Procesamiento local

- Emplea información del vecindario más próximo a cada punto etiquetado como frontera por el detector.
- Todos los puntos considerados *similares* según un criterio previo son enlazados.
- Normalmente, el criterio se construye a partir de la magnitud y la orientación del vector gradiente de cada punto.

- Dado un entorno S_{xy} de un píxel en (x, y), se considera que un punto $(s, t) \in S_{xy}$ es *similar* en magnitud al píxel (x, y) si:

$$|M(s,t) - M(x,y)| \leq \varepsilon$$

  donde $M(\cdot)$ es la magnigud del vector gradiente, y ε es un umbral positivo.
- Análogamente, se puede definir una similaridad en ángulo.

Enlazado de fronteras

- Cuando un punto situado en el entorno de un píxel *frontera* satisface los criterios de magnitud y ángulo, es marcado a su vez como frontera.

Enlazado de fronteras

- Cuando un punto situado en el entorno de un píxel *frontera* satisface los criterios de magnitud y ángulo, es marcado a su vez como frontera.
- El proceso se repite hasta que no se puedan marcar más puntos.

Enlazado de fronteras

- Cuando un punto situado en el entorno de un píxel *frontera* satisface los criterios de magnitud y ángulo, es marcado a su vez como frontera.
- El proceso se repite hasta que no se puedan marcar más puntos.
- Un método computacionalmente más eficiente puede ser el siguiente:

Enlazado de fronteras

- Cuando un punto situado en el entorno de un píxel *frontera* satisface los criterios de magnitud y ángulo, es marcado a su vez como frontera.
- El proceso se repite hasta que no se puedan marcar más puntos.
- Un método computacionalmente más eficiente puede ser el siguiente:
  1. Dado un umbral de magnitud T_M, un ángulo A, y un umbral de ángulo T_A, se crea una imagen binaria con los puntos que cumplen:

$$M(x,y) \geq T_M \text{ y } \alpha(x,y) = A \pm T_A$$

Enlazado de fronteras

- Cuando un punto situado en el entorno de un píxel *frontera* satisface los criterios de magnitud y ángulo, es marcado a su vez como frontera.
- El proceso se repite hasta que no se puedan marcar más puntos.
- Un método computacionalmente más eficiente puede ser el siguiente:
  1. Dado un umbral de magnitud T_M, un ángulo A, y un umbral de ángulo T_A, se crea una imagen binaria con los puntos que cumplen:

$$M(x,y) \geq T_M \text{ y } \alpha(x,y) = A \pm T_A$$

  2. Se recorre la imagen resultante por filas.

Enlazado de fronteras

- Cuando un punto situado en el entorno de un píxel *frontera* satisface los criterios de magnitud y ángulo, es marcado a su vez como frontera.
- El proceso se repite hasta que no se puedan marcar más puntos.
- Un método computacionalmente más eficiente puede ser el siguiente:
  1. Dado un umbral de magnitud T_M, un ángulo A, y un umbral de ángulo T_A, se crea una imagen binaria con los puntos que cumplen:

$$M(x,y) \geq T_M \text{ y } \alpha(x,y) = A \pm T_A$$

  2. Se recorre la imagen resultante por filas.
  3. Se rellenan los huecos más cortos que un umbral.

Enlazado de fronteras

- Cuando un punto situado en el entorno de un píxel *frontera* satisface los criterios de magnitud y ángulo, es marcado a su vez como frontera.
- El proceso se repite hasta que no se puedan marcar más puntos.
- Un método computacionalmente más eficiente puede ser el siguiente:
  1. Dado un umbral de magnitud T_M, un ángulo A, y un umbral de ángulo T_A, se crea una imagen binaria con los puntos que cumplen:

$$M(x,y) \geq T_M \text{ y } \alpha(x,y) = A \pm T_A$$

  2. Se recorre la imagen resultante por filas.
  3. Se rellenan los huecos más cortos que un umbral.
  4. Para otras orientaciones, se gira la imagen y se repite el proceso.

Ejemplo

a) Imagen original. b) Imagen gradiente. c) Fronteras horizontales. d) Fronteras verticales. e) OR de c) y d)

Procesamiento a nivel de región

- A veces se sabe a qué región pertenecen determinados puntos frontera.

Procesamiento a nivel de región

- A veces se sabe a qué región pertenecen determinados puntos frontera.
- Pueden enlazarse las fronteras mediante una aproximación poligonal.

Procesamiento a nivel de región

- A veces se sabe a qué región pertenecen determinados puntos frontera.
- Pueden enlazarse las fronteras mediante una aproximación poligonal.
- Se obtiene una representación compacta del contorno de la región, conservando sus características esenciales.

Procesamiento a nivel de región

- A veces se sabe a qué región pertenecen determinados puntos frontera.
- Pueden enlazarse las fronteras mediante una aproximación poligonal.
- Se obtiene una representación compacta del contorno de la región, conservando sus características esenciales.

- Estudiaremos primero el procedimiento de aproximación poligonal a un conjunto de puntos.

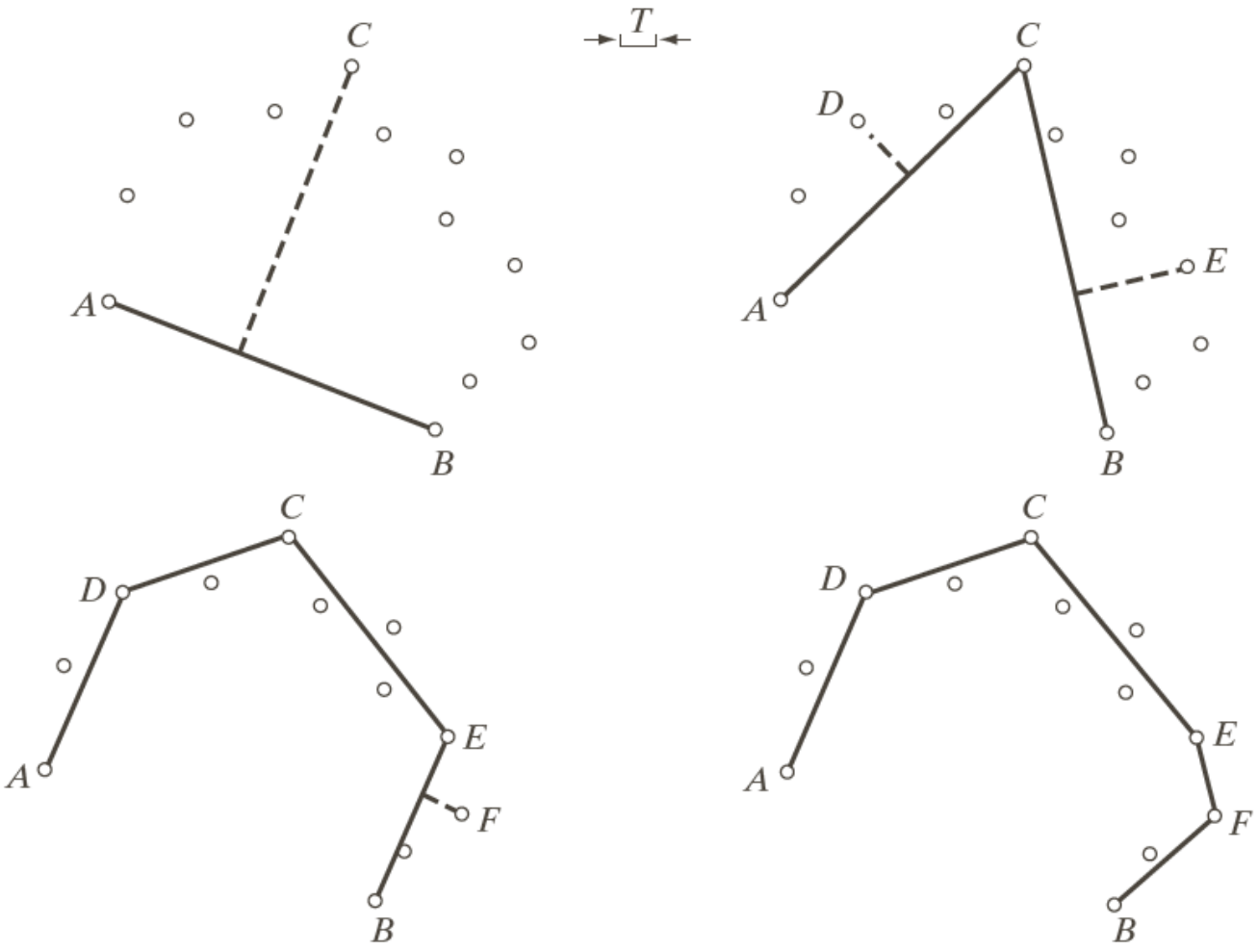
Procesamiento a nivel de región

- A veces se sabe a qué región pertenecen determinados puntos frontera.
- Pueden enlazarse las fronteras mediante una aproximación poligonal.
- Se obtiene una representación compacta del contorno de la región, conservando sus características esenciales.

- Estudiaremos primero el procedimiento de aproximación poligonal a un conjunto de puntos.
- Supondremos que los puntos están ordenados, y que conocemos los extremos (A y B).

Aproximación poligonal de un conjunto de puntos

Algoritmo de Douglas-Peucker:



Aproximación a curvas cerradas

- Tenemos el problema de que no hay punto inicial y final.

Aproximación a curvas cerradas

- Tenemos el problema de que no hay punto inicial y final.
- Si el conjunto de puntos está ordenado, podemos analizar las distancias entre puntos consecutivos. Si hay alguna superior a un umbral, podemos suponer la curva abierta en ese par de puntos.

Aproximación a curvas cerradas

- Tenemos el problema de que no hay punto inicial y final.
- Si el conjunto de puntos está ordenado, podemos analizar las distancias entre puntos consecutivos. Si hay alguna superior a un umbral, podemos suponer la curva abierta en ese par de puntos.
- Si la curva es cerrada, podemos coger como inicio el punto más a la derecha y como final el más a la izquierda (o viceversa).

Enlazado de píxeles sin información regional

- A veces tenemos una imagen de fronteras sin información acerca de dónde pueden estar las regiones de interés.

Enlazado de píxeles sin información regional

- A veces tenemos una imagen de fronteras sin información acerca de dónde pueden estar las regiones de interés.
- Los píxeles frontera solo pueden ser enlazados a partir de propiedades *globales*.

Enlazado de píxeles sin información regional

- A veces tenemos una imagen de fronteras sin información acerca de dónde pueden estar las regiones de interés.
- Los píxeles frontera solo pueden ser enlazados a partir de propiedades *globales*.
- Usaremos una aproximación que permite enlazar píxeles pertenecientes a curvas con una forma determinada.

Enlazado de píxeles sin información regional

- A veces tenemos una imagen de fronteras sin información acerca de dónde pueden estar las regiones de interés.
- Los píxeles frontera solo pueden ser enlazados a partir de propiedades *globales*.
- Usaremos una aproximación que permite enlazar píxeles pertenecientes a curvas con una forma determinada.

Ejemplo:

- Dados n puntos de una imagen, determinar conjuntos de los mismos que pertenezcan a líneas rectas.

Enlazado de píxeles sin información regional

- A veces tenemos una imagen de fronteras sin información acerca de dónde pueden estar las regiones de interés.
- Los píxeles frontera solo pueden ser enlazados a partir de propiedades *globales*.
- Usaremos una aproximación que permite enlazar píxeles pertenecientes a curvas con una forma determinada.

Ejemplo:

- Dados n puntos de una imagen, determinar conjuntos de los mismos que pertenezcan a líneas rectas.
- Una aproximación puede consistir en tomar pares de puntos, calcular la recta que los une, y buscar todos los demás puntos cercanos a esa recta.

Enlazado de píxeles sin información regional

- A veces tenemos una imagen de fronteras sin información acerca de dónde pueden estar las regiones de interés.
- Los píxeles frontera solo pueden ser enlazados a partir de propiedades *globales*.
- Usaremos una aproximación que permite enlazar píxeles pertenecientes a curvas con una forma determinada.

Ejemplo:

- Dados n puntos de una imagen, determinar conjuntos de los mismos que pertenezcan a líneas rectas.
- Una aproximación puede consistir en tomar pares de puntos, calcular la recta que los une, y buscar todos los demás puntos cercanos a esa recta.
- Esta aproximación es computacionalmente muy costosa.

Enlazado de píxeles sin información regional

- A veces tenemos una imagen de fronteras sin información acerca de dónde pueden estar las regiones de interés.
- Los píxeles frontera solo pueden ser enlazados a partir de propiedades *globales*.
- Usaremos una aproximación que permite enlazar píxeles pertenecientes a curvas con una forma determinada.

Ejemplo:

- Dados n puntos de una imagen, determinar conjuntos de los mismos que pertenezcan a líneas rectas.
- Una aproximación puede consistir en tomar pares de puntos, calcular la recta que los une, y buscar todos los demás puntos cercanos a esa recta.
- Esta aproximación es computacionalmente muy costosa.
- Hough (1962) propuso una aproximación alternativa, más eficiente.

La Transformada de Hough

Definición:

- Sea un punto (x_i, y_i) en un plano. Por él pasan infinitas rectas del tipo $y = ax + b$, siempre que sus parámetros (a, b) cumplan que $y_i = ax_i + b$.

La Transformada de Hough

Definición:

- Sea un punto (x_i, y_i) en un plano. Por él pasan infinitas rectas del tipo $y = ax + b$, siempre que sus parámetros (a, b) cumplan que $y_i = ax_i + b$.
- Escribiendo la ecuación anterior como $b = -x_i a + y_i$ y considerando el plano ab, cada par (x_i, y_i) da lugar a una recta en dicho plano.

La Transformada de Hough

Definición:

- Sea un punto (x_i, y_i) en un plano. Por él pasan infinitas rectas del tipo $y = ax + b$, siempre que sus parámetros (a, b) cumplan que $y_i = ax_i + b$.
- Escribiendo la ecuación anterior como $b = -x_i a + y_i$ y considerando el plano ab, cada par (x_i, y_i) da lugar a una recta en dicho plano.
- Dos puntos en xy generan dos rectas en ab. Dichas rectas se cortarán en un punto del plano ab que se corresponde con los parámetros de la recta que une los dos puntos iniciales en xy.

La Transformada de Hough

Definición:

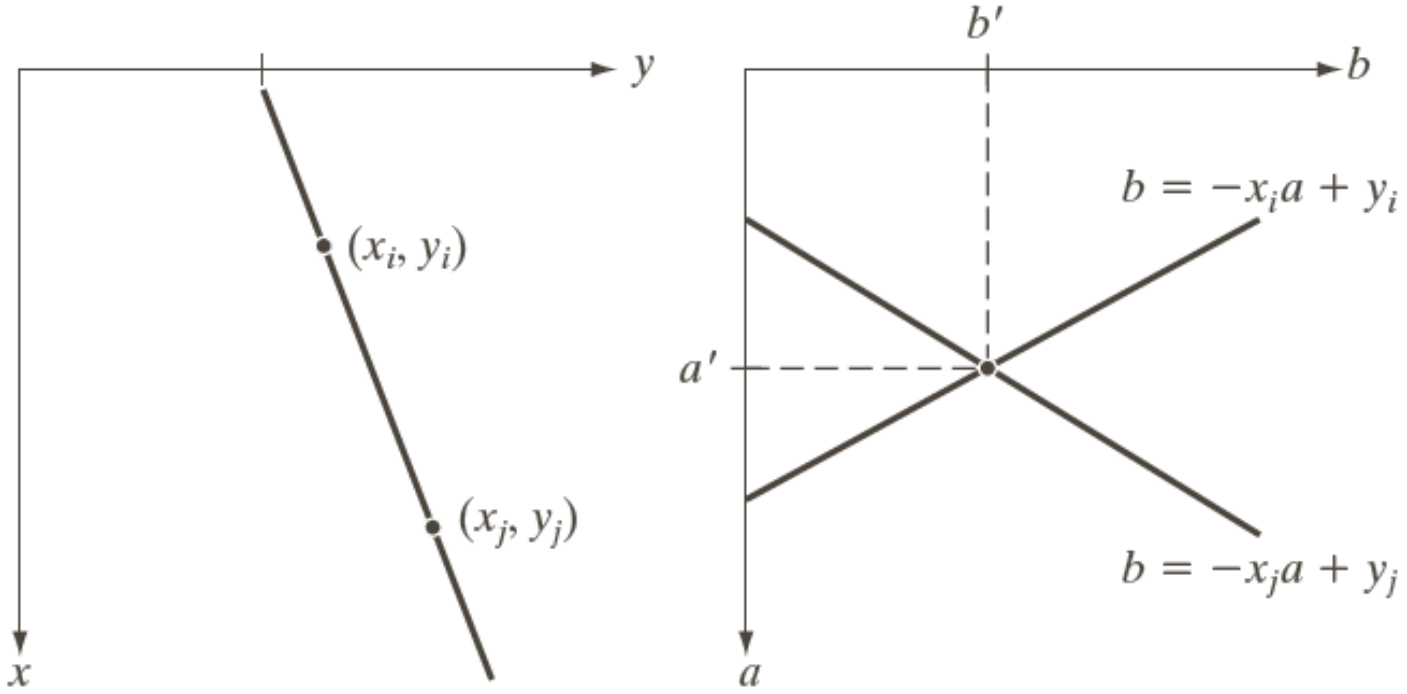
- Sea un punto (x_i, y_i) en un plano. Por él pasan infinitas rectas del tipo $y = ax + b$, siempre que sus parámetros (a, b) cumplan que $y_i = ax_i + b$.
- Escribiendo la ecuación anterior como $b = -x_i a + y_i$ y considerando el plano ab, cada par (x_i, y_i) da lugar a una recta en dicho plano.
- Dos puntos en xy generan dos rectas en ab. Dichas rectas se cortarán en un punto del plano ab que se corresponde con los parámetros de la recta que une los dos puntos iniciales en xy.
- Cada punto en el plano ab caracteriza una recta distinta en xy.

La Transformada de Hough

Definición:

- Sea un punto (x_i, y_i) en un plano. Por él pasan infinitas rectas del tipo $y = ax + b$, siempre que sus parámetros (a, b) cumplan que $y_i = ax_i + b$.
- Escribiendo la ecuación anterior como $b = -x_i a + y_i$ y considerando el plano ab, cada par (x_i, y_i) da lugar a una recta en dicho plano.
- Dos puntos en xy generan dos rectas en ab. Dichas rectas se cortarán en un punto del plano ab que se corresponde con los parámetros de la recta que une los dos puntos iniciales en xy.
- Cada punto en el plano ab caracteriza una recta distinta en xy.
- Todos los puntos de una recta en xy generan rectas en ab que se cruzan en un mismo punto.

Espacio transformado



Inconvenientes del Espacio Transformado de Hough

- Cuando una linea se aproxima a la vertical, el valor de su pendiente (a) se aproxima a infinito.

Inconvenientes del Espacio Transformado de Hough

- Cuando una linea se aproxima a la vertical, el valor de su pendiente (a) se aproxima a infinito.
- Una forma de evitar esto es emplear la siguiente representación para una línea:

$$x \cos\theta + y \sin\theta = \rho$$

Inconvenientes del Espacio Transformado de Hough

- Cuando una linea se aproxima a la vertical, el valor de su pendiente (a) se aproxima a infinito.
- Una forma de evitar esto es emplear la siguiente representación para una línea:

$$x \cos\theta + y \sin\theta = \rho$$

- Una línea horizontal tiene $\theta = 90°$.

Inconvenientes del Espacio Transformado de Hough

- Cuando una linea se aproxima a la vertical, el valor de su pendiente (a) se aproxima a infinito.
- Una forma de evitar esto es emplear la siguiente representación para una línea:

$$x \cos\theta + y \sin\theta = \rho$$

- Una línea horizontal tiene $\theta = 90°$.
- En este espacio, la familia de líneas que pasan por un punto forma una curva sinusoidal.

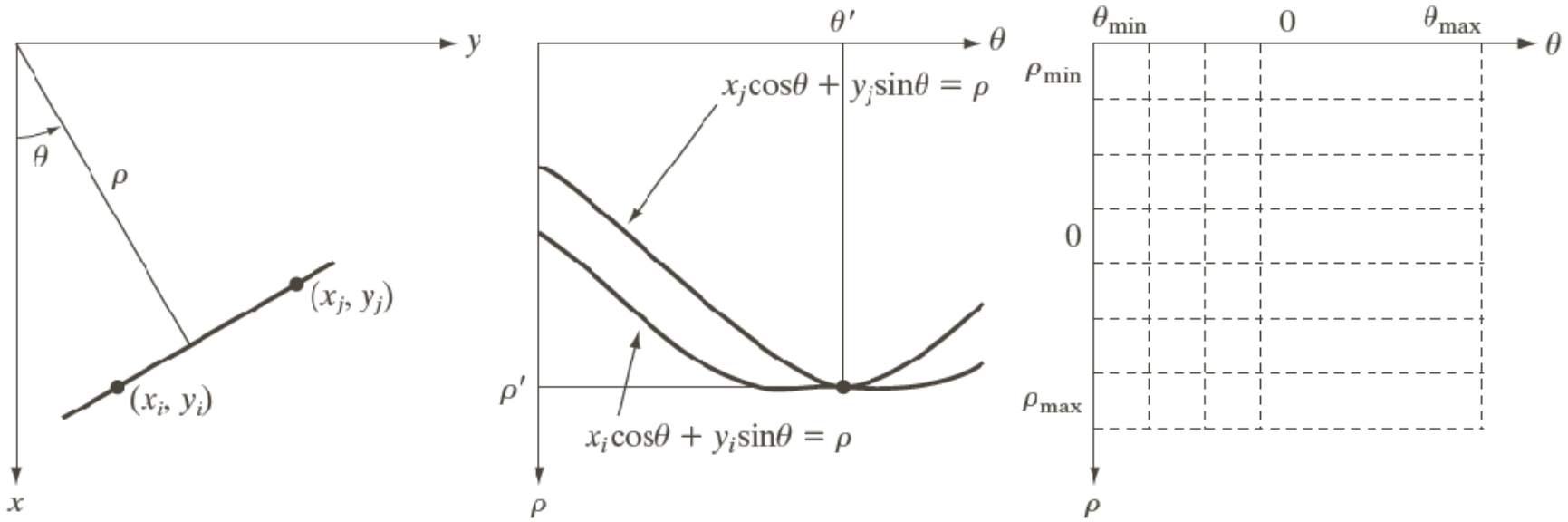
Inconvenientes del Espacio Transformado de Hough

- Cuando una linea se aproxima a la vertical, el valor de su pendiente (a) se aproxima a infinito.
- Una forma de evitar esto es emplear la siguiente representación para una línea:

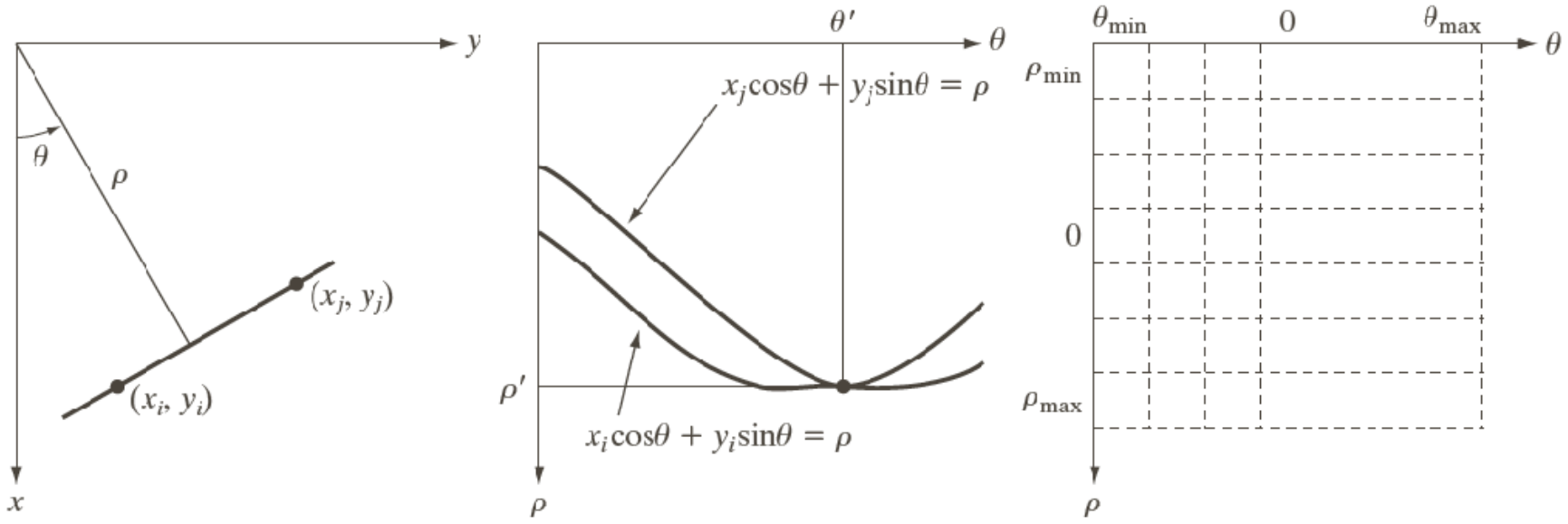
$$x \cos\theta + y \sin\theta = \rho$$

- Una línea horizontal tiene $\theta = 90°$.
- En este espacio, la familia de líneas que pasan por un punto forma una curva sinusoidal.
- Podemos subdividir el espacio $\rho\theta$ en *celdas acumuladoras*.

Espacio transformado

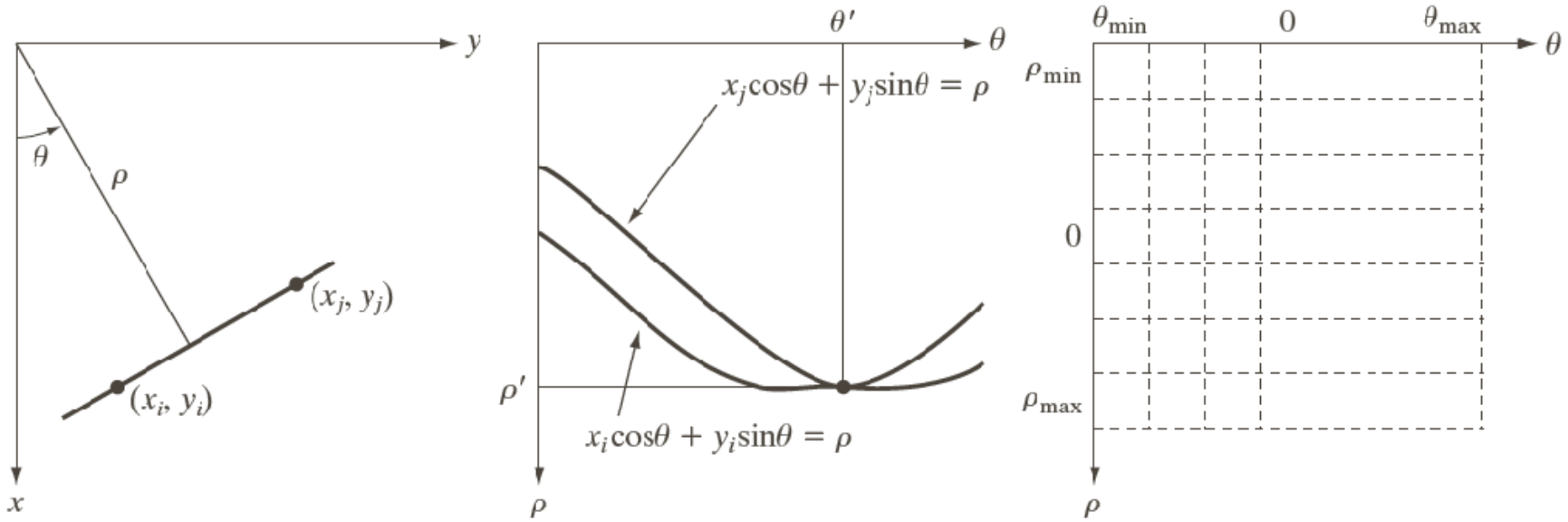


Espacio transformado



- Este método puede generalizarse a cualquier función del tipo $g(\mathrm{v}, \mathrm{c}) = 0$, donde v es un vector de coordenadas, y c es un vector de coeficientes.

Espacio transformado

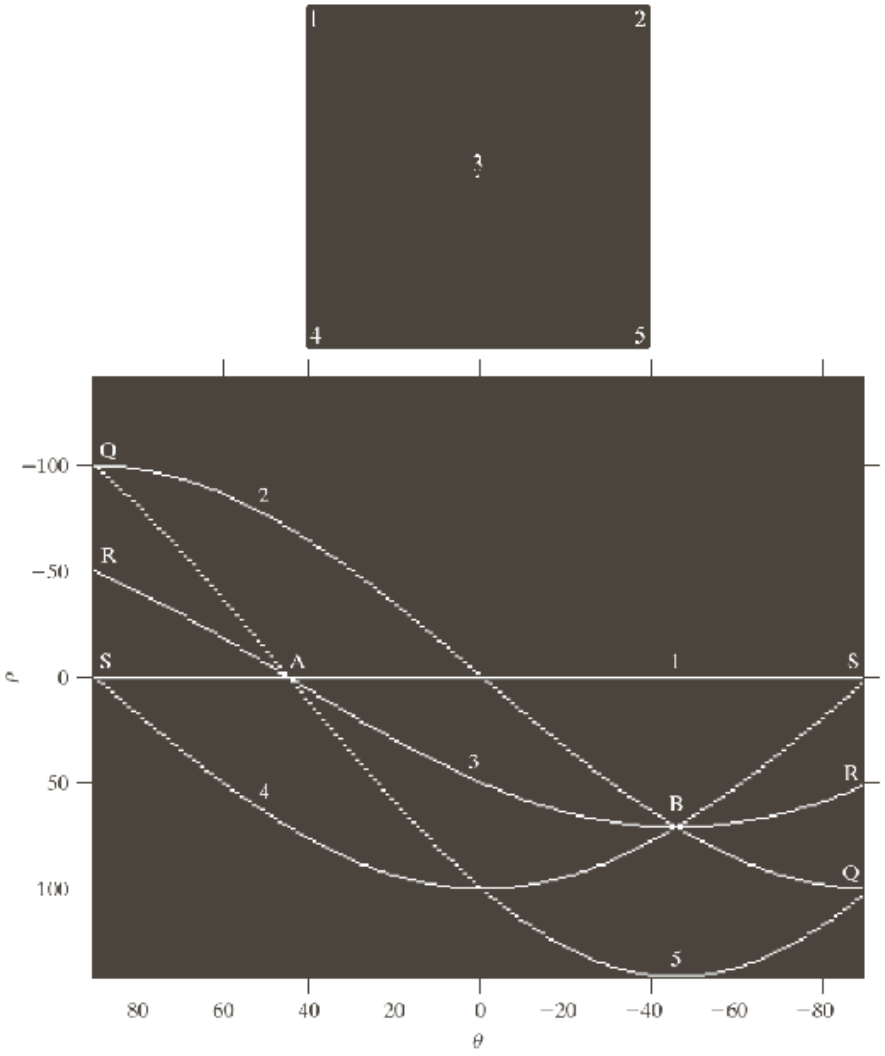


- Este método puede generalizarse a cualquier función del tipo $g(\mathrm{v}, \mathrm{c}) = 0$, donde v es un vector de coordenadas, y c es un vector de coeficientes.
- Por ejemplo, si queremos detectar circunferencias usaremos:

$$(x - c_1)^2 + (y - c_2)^2 = c_3^2$$

Obteniendo un espacio transformado tridimensional.

Ejemplo



Segmentación mediante umbralización

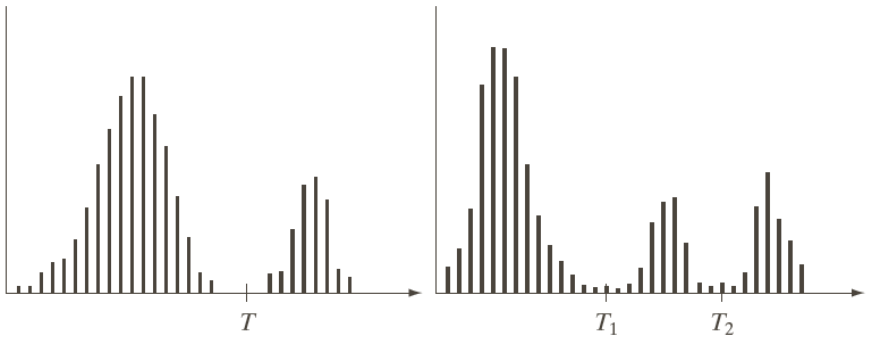
- La umbralización *(thresholding)* puede ser usada para dividir la imagen en regiones con valores de intensidad (o alguna otra propiedad) similares.

Segmentación mediante umbralización

- La umbralización *(thresholding)* puede ser usada para dividir la imagen en regiones con valores de intensidad (o alguna otra propiedad) similares.
- Una imagen con objetos brillantes sobre fondo oscuro tendrá un histograma como los de la figura.

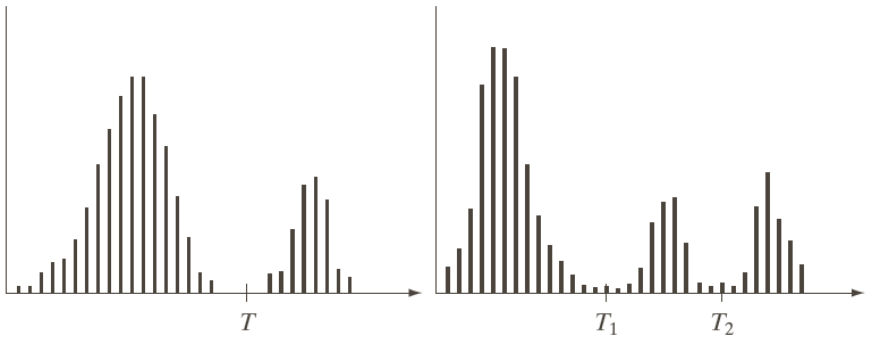
Segmentación mediante umbralización

- La umbralización *(thresholding)* puede ser usada para dividir la imagen en regiones con valores de intensidad (o alguna otra propiedad) similares.
- Una imagen con objetos brillantes sobre fondo oscuro tendrá un histograma como los de la figura.



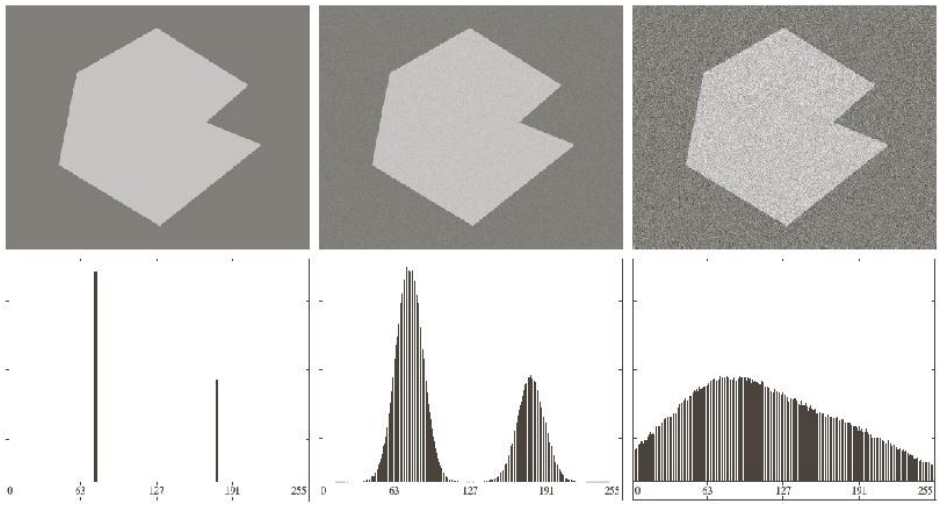
Segmentación mediante umbralización

- La umbralización *(thresholding)* puede ser usada para dividir la imagen en regiones con valores de intensidad (o alguna otra propiedad) similares.
- Una imagen con objetos brillantes sobre fondo oscuro tendrá un histograma como los de la figura.
- Puede ser posible segmentar la imagen si encontramos un valor global T que separe las modas del histograma.



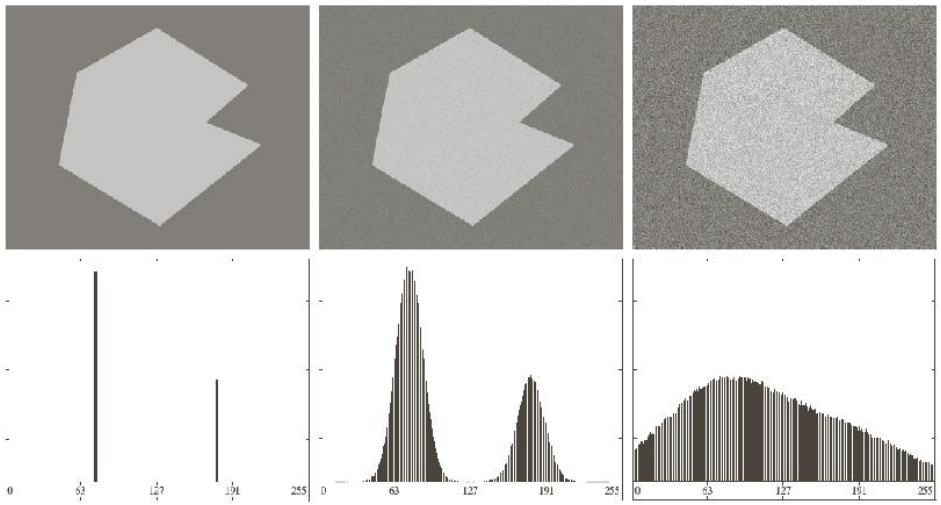
Umbralización en imágenes con ruido

- En algunos casos, puede resultar muy difícil separar las modas del histograma.



Umbralización en imágenes con ruido

- En algunos casos, puede resultar muy difícil separar las modas del histograma.
- Cuando aumenta el nivel de ruido, las modas del histograma se mezclan.



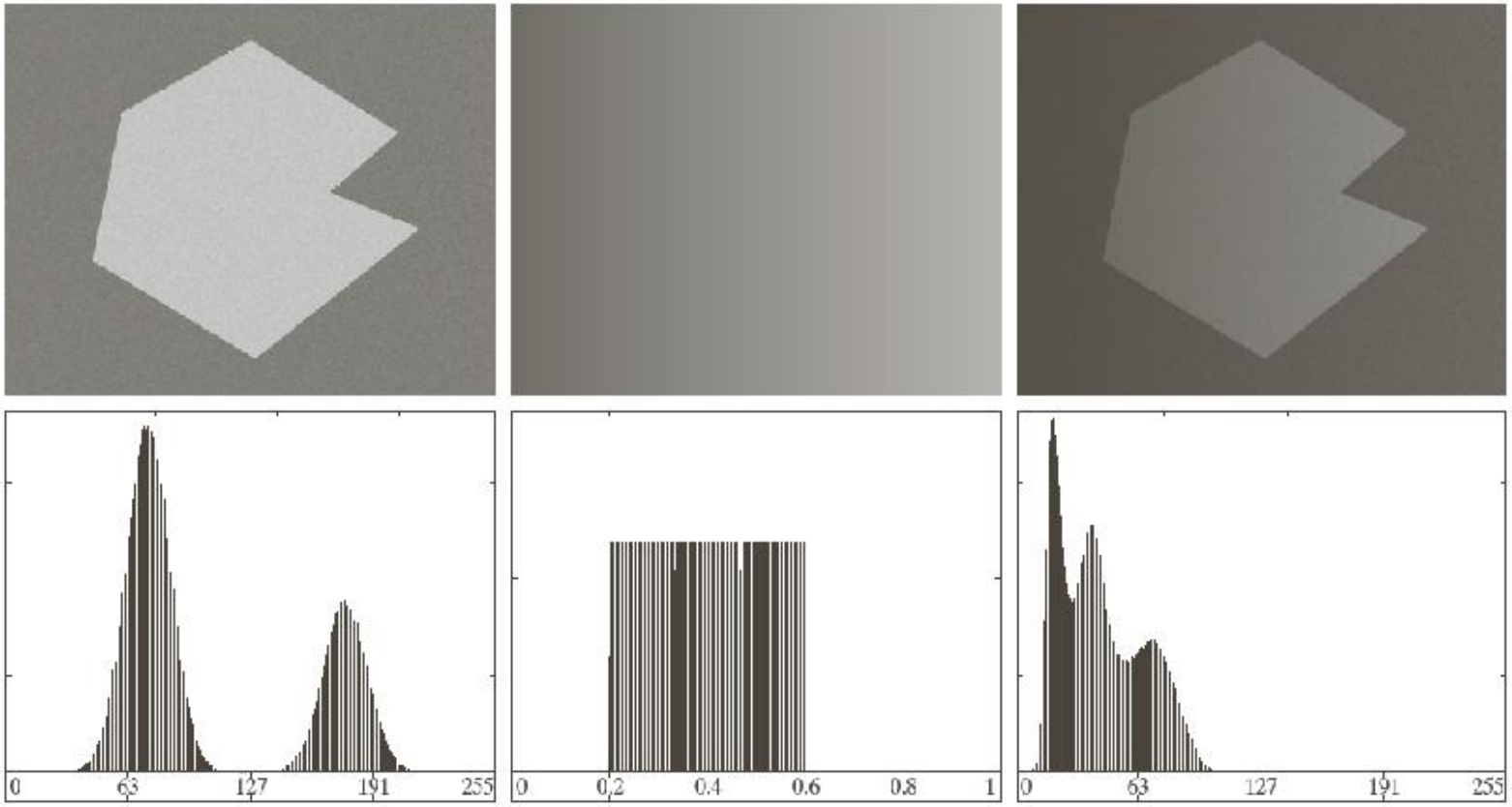
Umbralización global frente a dinámica

- En muchos casos, encontrar un único umbral para toda la imagen es imposible.
  - Cambios de iluminación, reflectancia, ruido, etc.

Umbralización global frente a dinámica

- En muchos casos, encontrar un único umbral para toda la imagen es imposible.
  - Cambios de iluminación, reflectancia, ruido, etc.
- El umbral T, en lugar de ser único, puede variar en función de la posición dentro de la imagen, dando lugar a métodos *variables, dinámicos o adaptativos* de umbralización.

Ejemplo de iluminación variable



Umbralización global básica

- El siguiente algoritmo iterativo permite calcular un umbral:

Umbralización global básica

- El siguiente algoritmo iterativo permite calcular un umbral:
  1. Seleccionar un umbral inicial T.
  2. Segmentar la imagen a partir de dicho umbral. G_1 es el área con intensidad $< T$, y G_2 el resto.
  3. Calcular la intensidad media de G_1 y G_2, m_1 y m_2.
  4. Actualizar el valor de T:

$$T = \frac{1}{2}(m_1 + m_2)$$

  5. Repetir los pasos 2 a 4 hasta que el valor de T se estabilice.

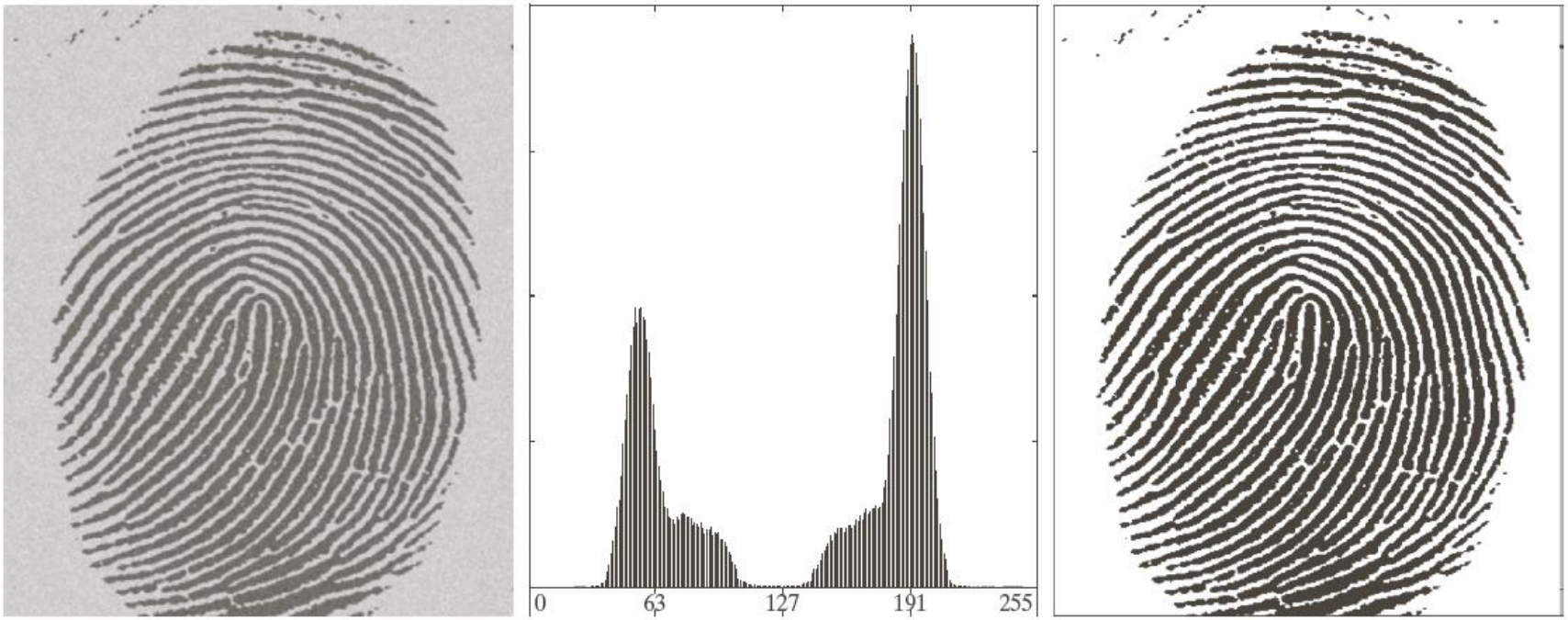
Umbralización global básica

- El siguiente algoritmo iterativo permite calcular un umbral:
  1. Seleccionar un umbral inicial T.
  2. Segmentar la imagen a partir de dicho umbral. G_1 es el área con intensidad $< T$, y G_2 el resto.
  3. Calcular la intensidad media de G_1 y G_2, m_1 y m_2.
  4. Actualizar el valor de T:

$$T = \frac{1}{2}(m_1 + m_2)$$

  5. Repetir los pasos 2 a 4 hasta que el valor de T se estabilice.
- Tanto m_1 como m_2 pueden calcularse directamente a partir del histograma.

Ejemplo de umbralización global



Método de estimación de umbral de Otsu

- Se basa en minimizar el error medio cometido al segmentar.

Método de estimación de umbral de Otsu

- Se basa en minimizar el error medio cometido al segmentar.
- Usa la regla de Bayes, suponiendo funciones de densidad de probabilidad gaussianas.

Método de estimación de umbral de Otsu

- Se basa en minimizar el error medio cometido al segmentar.
- Usa la regla de Bayes, suponiendo funciones de densidad de probabilidad gaussianas.
- Para una imagen G, con L tonos de gris, y siendo p_i los componentes del histograma normalizado ($\sum p_i = 1$), definimos para un umbral k:

$$P_1(k) = \sum_{i=0}^{k} p_i$$

Método de estimación de umbral de Otsu

- Se basa en minimizar el error medio cometido al segmentar.
- Usa la regla de Bayes, suponiendo funciones de densidad de probabilidad gaussianas.
- Para una imagen G, con L tonos de gris, y siendo p_i los componentes del histograma normalizado ($\sum p_i = 1$), definimos para un umbral k:

$$P_1(k) = \sum_{i=0}^{k} p_i$$

- Definimos también:

$$m(k) = \sum_{i=0}^{k} ip_i \qquad m_G = \sum_{i=0}^{L-1} ip_i$$

Método de estimación de umbral de Otsu (cont.)

- Buscamos el valor k que maximiza la expresión:

$$\frac{[m_G P_1(k) - m(k)]^2}{P_1(k)\,[1 - P_1(k)]}$$

Método de estimación de umbral de Otsu (cont.)

- Buscamos el valor k que maximiza la expresión:

$$\frac{[m_G P_1(k) - m(k)]^2}{P_1(k)\,[1 - P_1(k)]}$$

- El algoritmo permite evaluar también la *separabilidad* de las modas del histograma.

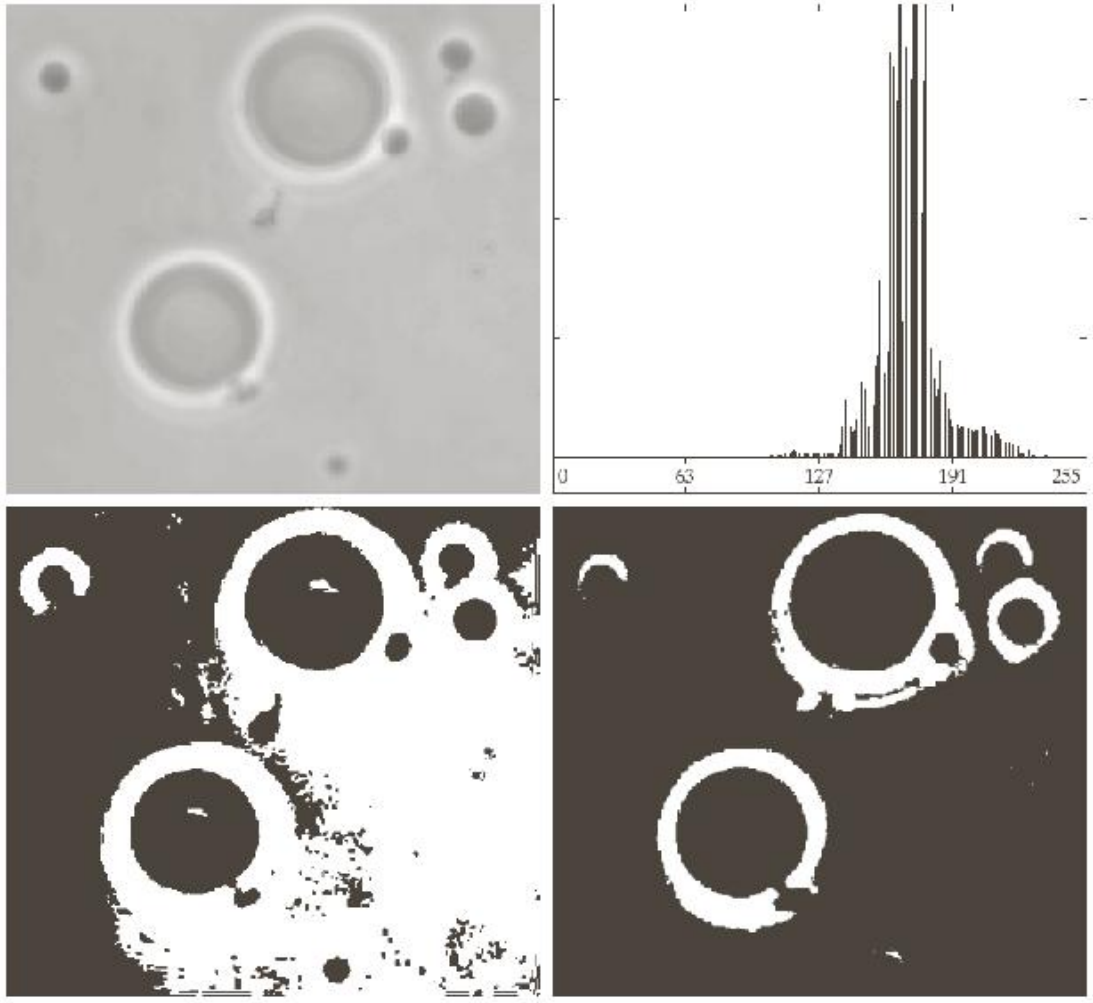
Método de estimación de umbral de Otsu (cont.)

- Buscamos el valor k que maximiza la expresión:

$$\frac{[m_G P_1(k) - m(k)]^2}{P_1(k)\,[1 - P_1(k)]}$$

- El algoritmo permite evaluar también la *separabilidad* de las modas del histograma.
- Este algoritmo calcula el umbral k de forma directa, por lo que es más rápido que el método básico, que es iterativo.

Ejemplo de umbralización de Otsu



Uso del suavizado para mejorar la umbralización

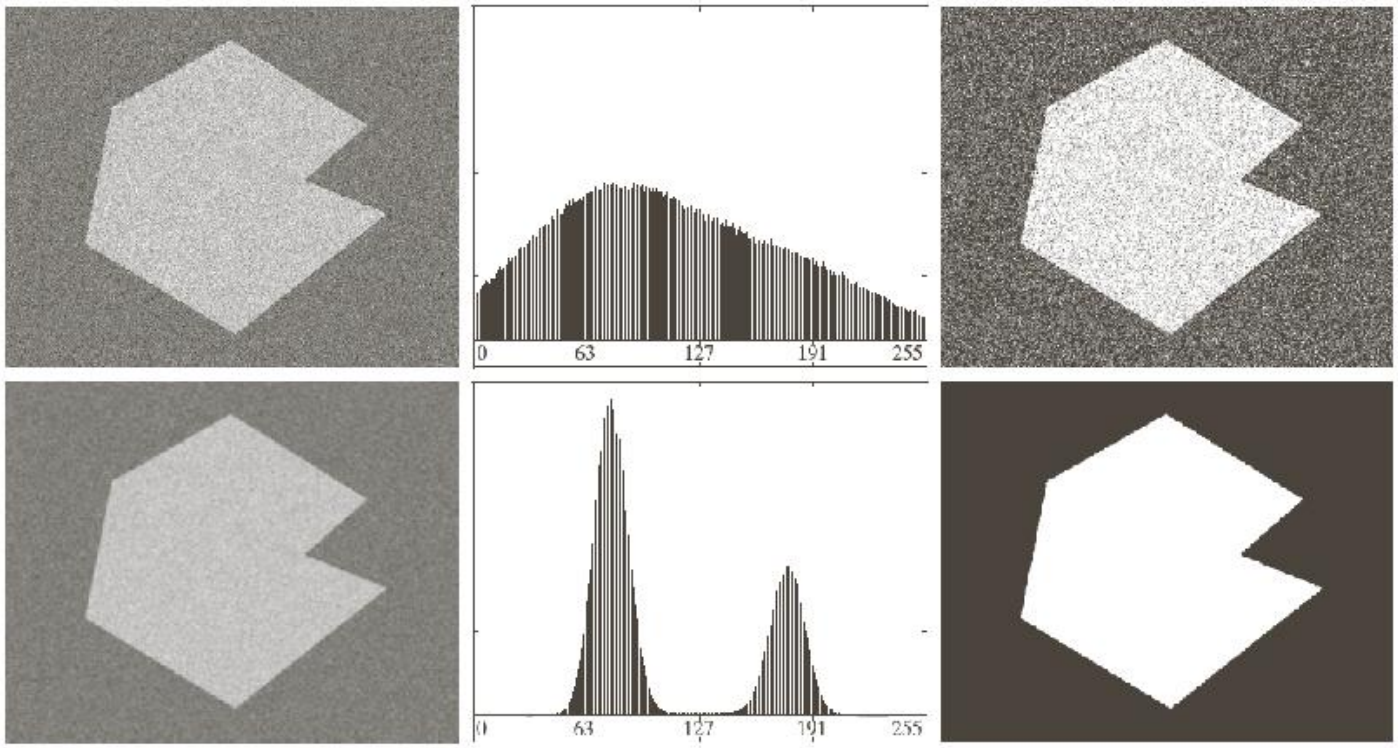
- En algunos casos, el ruido hace que sea imposible encontrar un umbral.

Uso del suavizado para mejorar la umbralización

- En algunos casos, el ruido hace que sea imposible encontrar un umbral.
- El suavizado puede ayudar a evitar ese problema.

Uso del suavizado para mejorar la umbralización

- En algunos casos, el ruido hace que sea imposible encontrar un umbral.
- El suavizado puede ayudar a evitar ese problema.



Umbralización y tamaño de los objetos

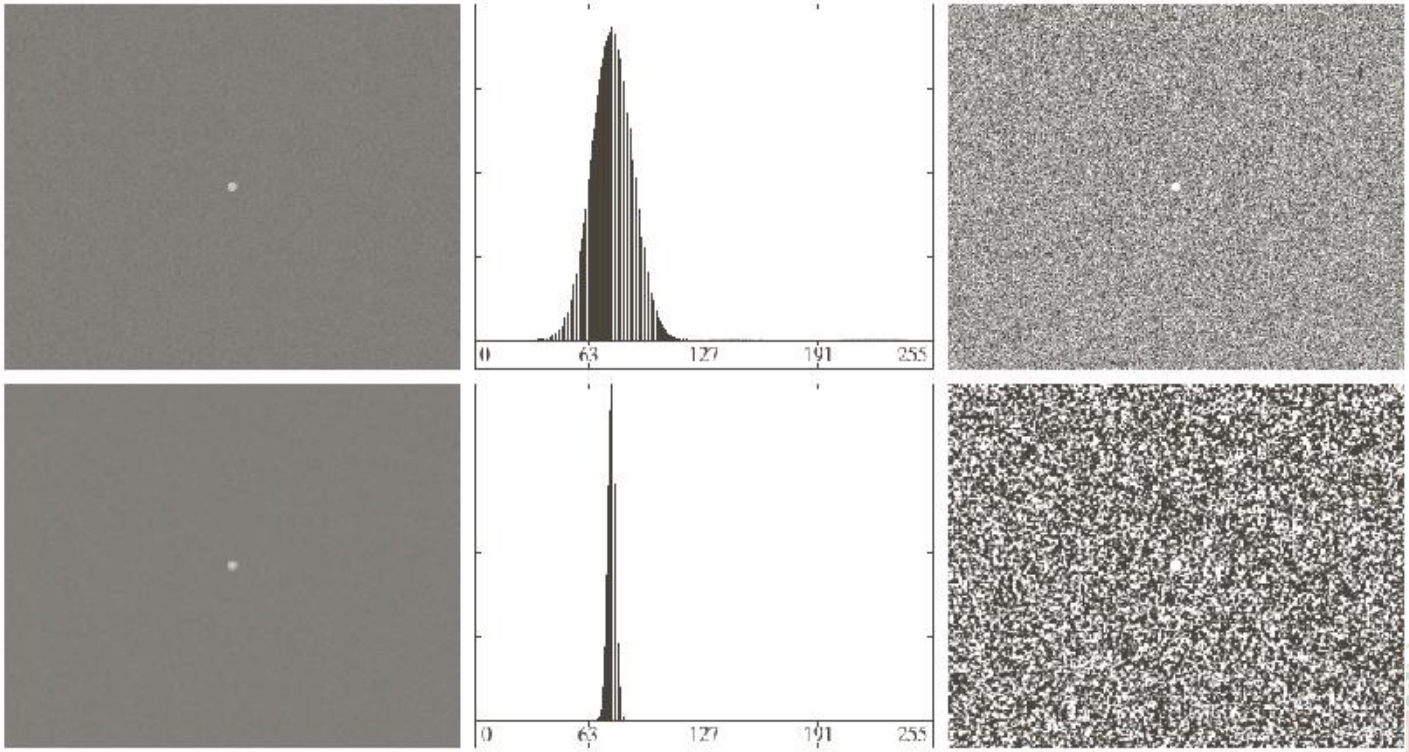
- Cuando hay áreas con tamaños muy dispares, las áreas mayores dominan el histograma.

Umbralización y tamaño de los objetos

- Cuando hay áreas con tamaños muy dispares, las áreas mayores dominan el histograma.
- Eso dificulta mucho el cálculo del umbral.

Umbralización y tamaño de los objetos

- Cuando hay áreas con tamaños muy dispares, las áreas mayores dominan el histograma.
- Eso dificulta mucho el cálculo del umbral.



Uso de fronteras en la umbralización

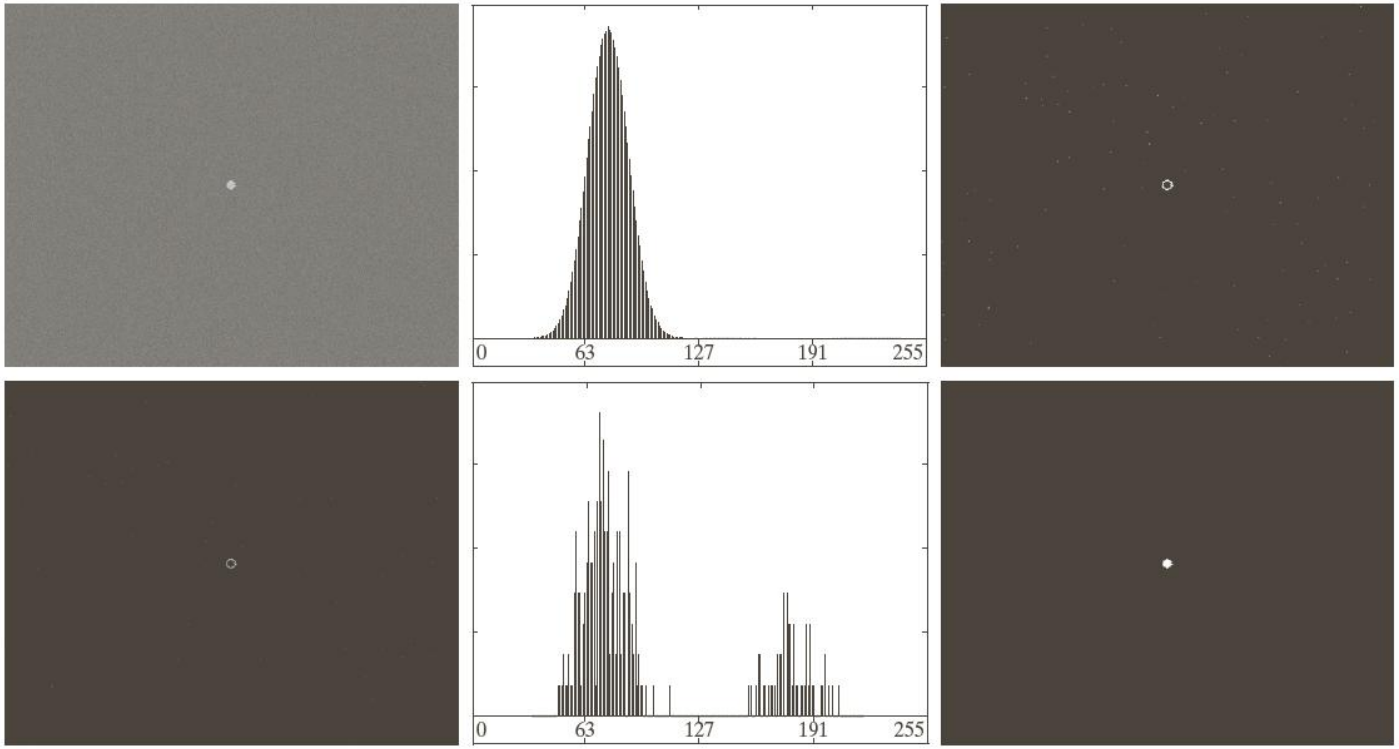
- Podemos crear una máscara binaria a partir de la magnitud de gradiente de la imagen.

Uso de fronteras en la umbralización

- Podemos crear una máscara binaria a partir de la magnitud de gradiente de la imagen.
- Los picos del histograma son más parecidos en tamaño.

Uso de fronteras en la umbralización

- Podemos crear una máscara binaria a partir de la magnitud de gradiente de la imagen.
- Los picos del histograma son más parecidos en tamaño.



Umbralización variable

- Permite calcular distintos umbrales para una misma imagen.

Umbralización variable

- Permite calcular distintos umbrales para una misma imagen.
- Tipos:

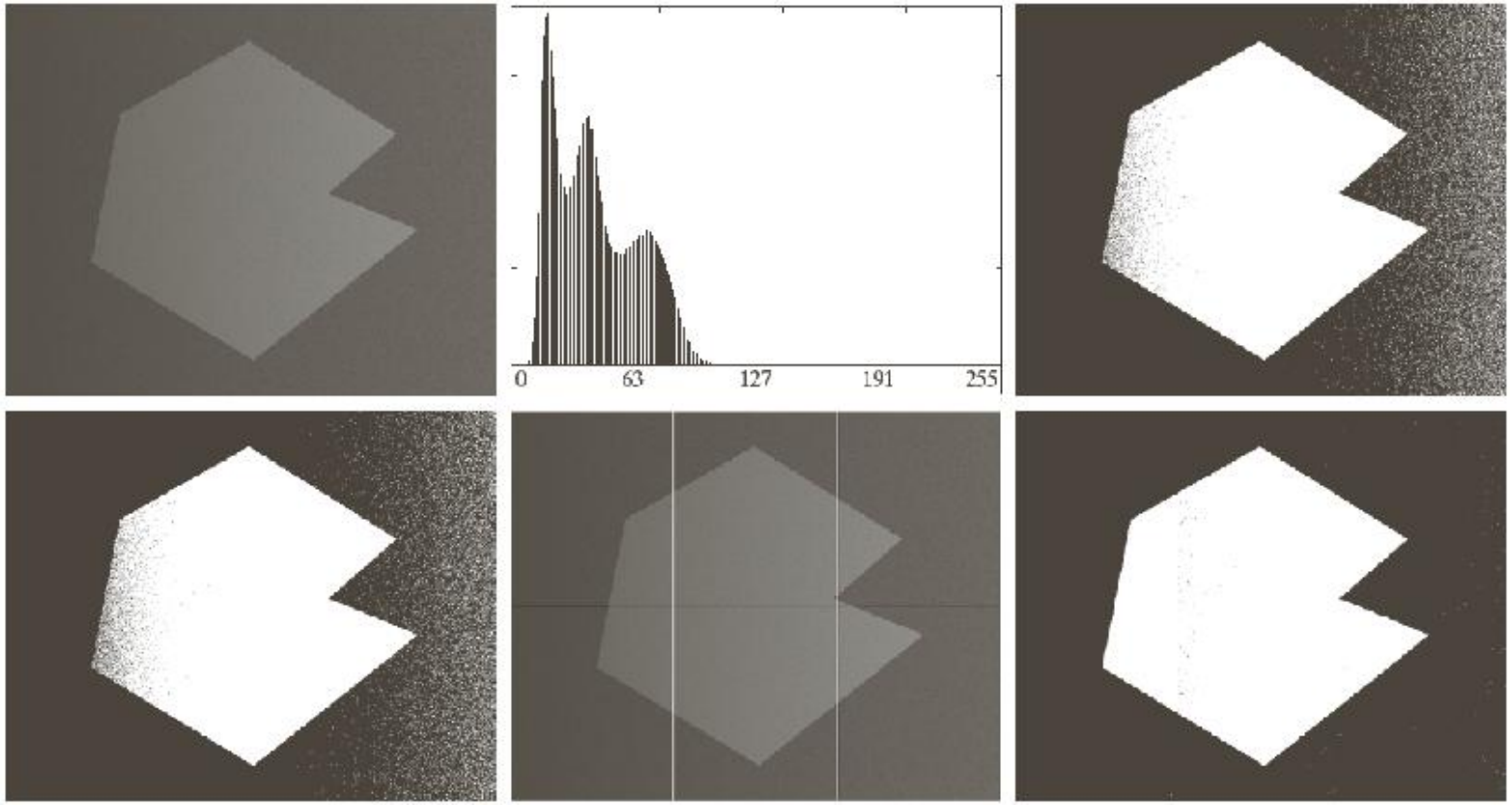
Umbralización variable

- Permite calcular distintos umbrales para una misma imagen.
- Tipos:
  - Particionado de la imagen.

Umbralización variable

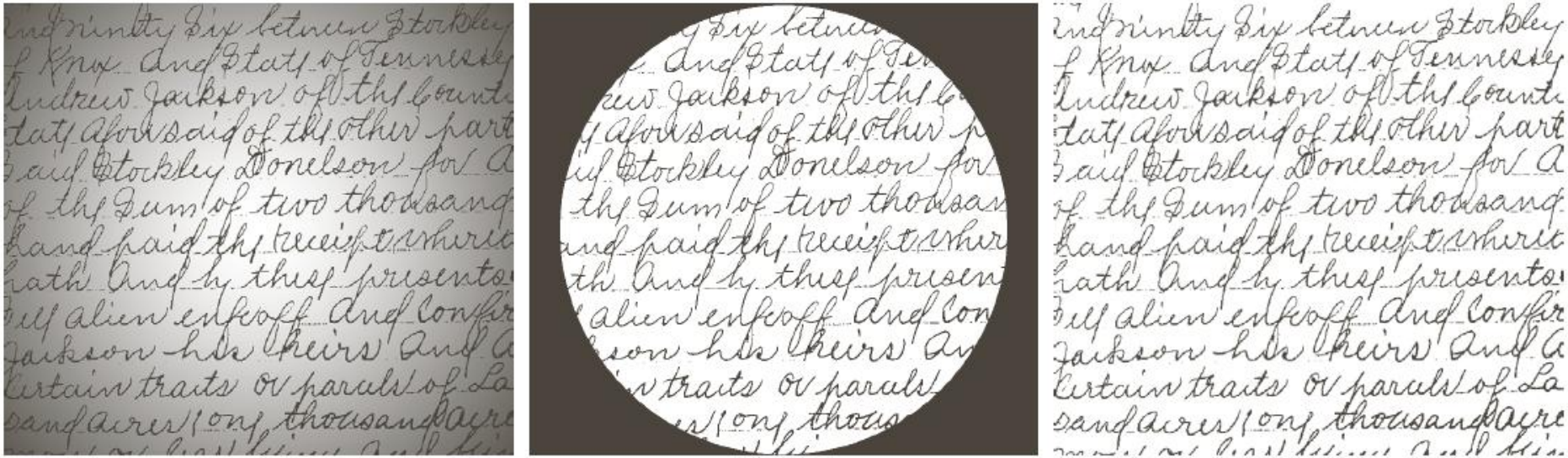
- Permite calcular distintos umbrales para una misma imagen.
- Tipos:
  - Particionado de la imagen.
  - Cálculo del umbral a partir del entorno de cada punto.

Umbralización mediante particionado



Umbralizados globales frente a umbralizado mediante partición de la imagen en seis trozos.

Umbralización mediante entornos locales



Centro: Umbralizado de Otsu; Derecha: Umbralizado mediante entornos locales.

Segmentación basada en regiones

- Las técnicas de umbralización no tienen en cuenta qué ocurre con los vecinos de un punto a la hora de asignarle una etiqueta.

Segmentación basada en regiones

- Las técnicas de umbralización no tienen en cuenta qué ocurre con los vecinos de un punto a la hora de asignarle una etiqueta.
- Por lo tanto, pueden salir regiones inconexas y puntos aislados.

Segmentación basada en regiones

- Las técnicas de umbralización no tienen en cuenta qué ocurre con los vecinos de un punto a la hora de asignarle una etiqueta.
- Por lo tanto, pueden salir regiones inconexas y puntos aislados.
- Existen métodos que buscan específicamente generar regiones:

Segmentación basada en regiones

- Las técnicas de umbralización no tienen en cuenta qué ocurre con los vecinos de un punto a la hora de asignarle una etiqueta.
- Por lo tanto, pueden salir regiones inconexas y puntos aislados.
- Existen métodos que buscan específicamente generar regiones:
  - Crecimiento de regiones.
  - Partición y mezcla.
  - Inundación.

Crecimiento de regiones

Consiste en agrupar píxeles o subregiones en otras más grandes basándose en un criterio de crecimiento predefinido.

Crecimiento de regiones

Consiste en agrupar píxeles o subregiones en otras más grandes basándose en un criterio de crecimiento predefinido.

- Se empieza con una o varias *semillas*.

Crecimiento de regiones

Consiste en agrupar píxeles o subregiones en otras más grandes basándose en un criterio de crecimiento predefinido.

- Se empieza con una o varias *semillas*.
- De acuerdo con algún criterio, se van añadiendo píxeles de entre los vecinos directos de cada región, hasta que no se puedan añadir más.

Crecimiento de regiones

Consiste en agrupar píxeles o subregiones en otras más grandes basándose en un criterio de crecimiento predefinido.

- Se empieza con una o varias *semillas*.
- De acuerdo con algún criterio, se van añadiendo píxeles de entre los vecinos directos de cada región, hasta que no se puedan añadir más.

Colocación de las semillas

- Si no se dispone de conocimiento *a priori*, se suelen evaluar las condiciones que dan lugar al crecimiento, y buscar agrupaciones de píxeles que las cumplan. Se toman como semillas los centroides de dichas regiones.

Algoritmo general de crecimiento de regiones

Sea $f(x, y)$ la imagen de entrada; $S(x, y)$ un array de semillas con valor 1 donde haya semillas y 0 en el resto; Q un predicado aplicable a cada píxel.

1. *Erosionar* el array S para que cada componente conexa quede reducida a un único píxel.

Algoritmo general de crecimiento de regiones

Sea $f(x, y)$ la imagen de entrada; $S(x, y)$ un array de semillas con valor 1 donde haya semillas y 0 en el resto; Q un predicado aplicable a cada píxel.

1. *Erosionar* el array S para que cada componente conexa quede reducida a un único píxel.
2. Formar una imagen f_Q que valga 1 en los puntos que satisfagan el predicado $Q(x, y)$ y 0 en el resto.

Algoritmo general de crecimiento de regiones

Sea $f(x, y)$ la imagen de entrada; $S(x, y)$ un array de semillas con valor 1 donde haya semillas y 0 en el resto; Q un predicado aplicable a cada píxel.

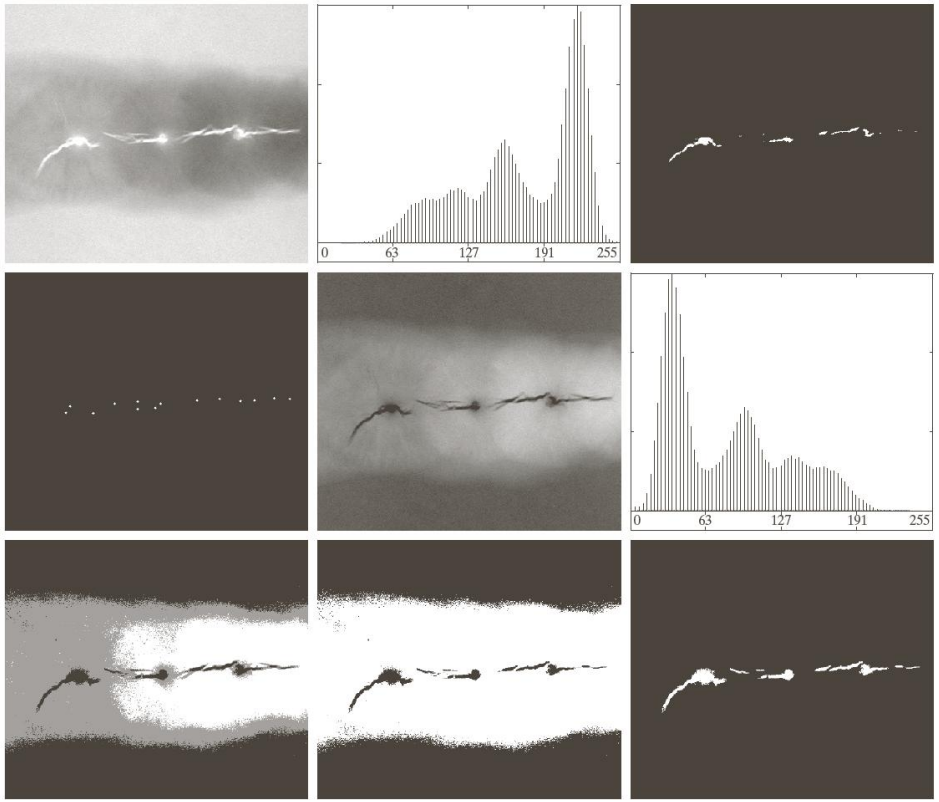
1. *Erosionar* el array S para que cada componente conexa quede reducida a un único píxel.
2. Formar una imagen f_Q que valga 1 en los puntos que satisfagan el predicado $Q(x, y)$ y 0 en el resto.
3. Sea g una imagen formada añadiendo a cada punto en S todos los puntos (x, y) tales que estén 8-conectados con algún punto semilla y $f_Q(x, y) = 1$.

Algoritmo general de crecimiento de regiones

Sea $f(x, y)$ la imagen de entrada; $S(x, y)$ un array de semillas con valor 1 donde haya semillas y 0 en el resto; Q un predicado aplicable a cada píxel.

1. *Erosionar* el array S para que cada componente conexa quede reducida a un único píxel.
2. Formar una imagen f_Q que valga 1 en los puntos que satisfagan el predicado $Q(x, y)$ y 0 en el resto.
3. Sea g una imagen formada añadiendo a cada punto en S todos los puntos (x, y) tales que estén 8-conectados con algún punto semilla y $f_Q(x, y) = 1$.
4. Etiquetar cada componente conexa en g con un valor diferente.

Ejemplo de crecimiento de regiones



- Predicado: que la diferencia entre la semilla y el punto sea menor que un umbral.
- Arriba derecha: semillas iniciales.
- Centro izquierda: semillas erosionadas.
- Abajo derecha: segmentación mediante crecimiento de regiones.

Partición y mezcla

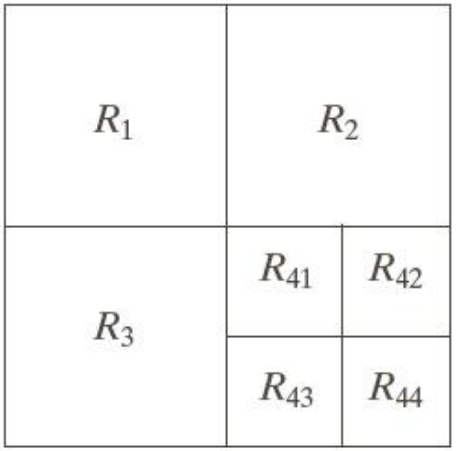
- Siendo R una imagen y Q un predicado, podemos subdividir R en cuadrantes sucesivamente hasta que todos los píxeles de cada región cumplan Q (o $\neg Q$).

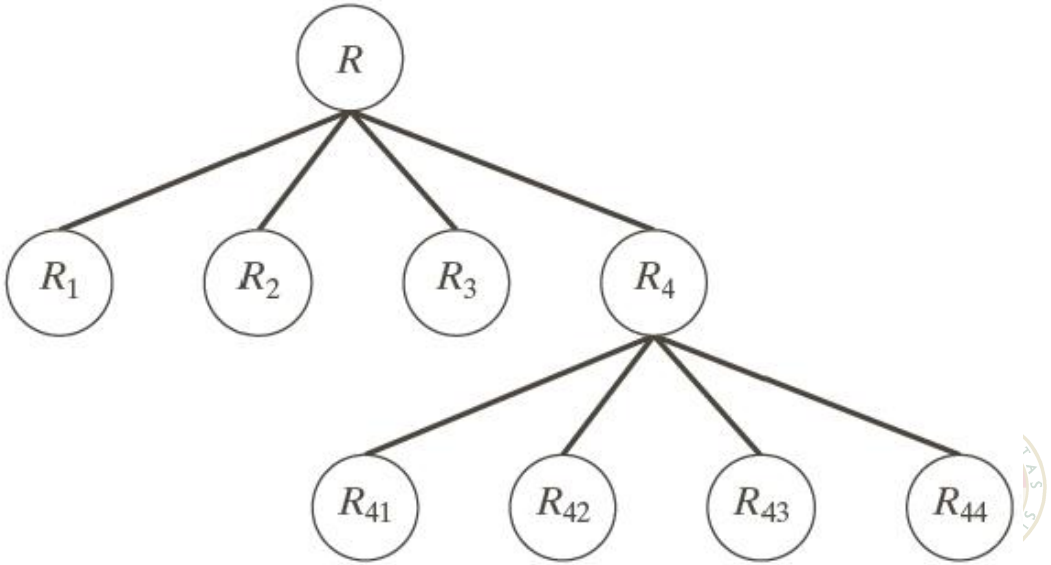
Partición y mezcla

- Siendo R una imagen y Q un predicado, podemos subdividir R en cuadrantes sucesivamente hasta que todos los píxeles de cada región cumplan Q (o $\neg Q$).
- Empezamos con la imagen entera. Si Q no es *Verdadero* (o *Falso)* para toda ella, la subdividimos en cuadrantes, y aplicamos el mismo criterio a cada uno de ellos.

Partición y mezcla

- Siendo R una imagen y Q un predicado, podemos subdividir R en cuadrantes sucesivamente hasta que todos los píxeles de cada región cumplan Q (o $\neg Q$).
- Empezamos con la imagen entera. Si Q no es *Verdadero* (o *Falso)* para toda ella, la subdividimos en cuadrantes, y aplicamos el mismo criterio a cada uno de ellos.





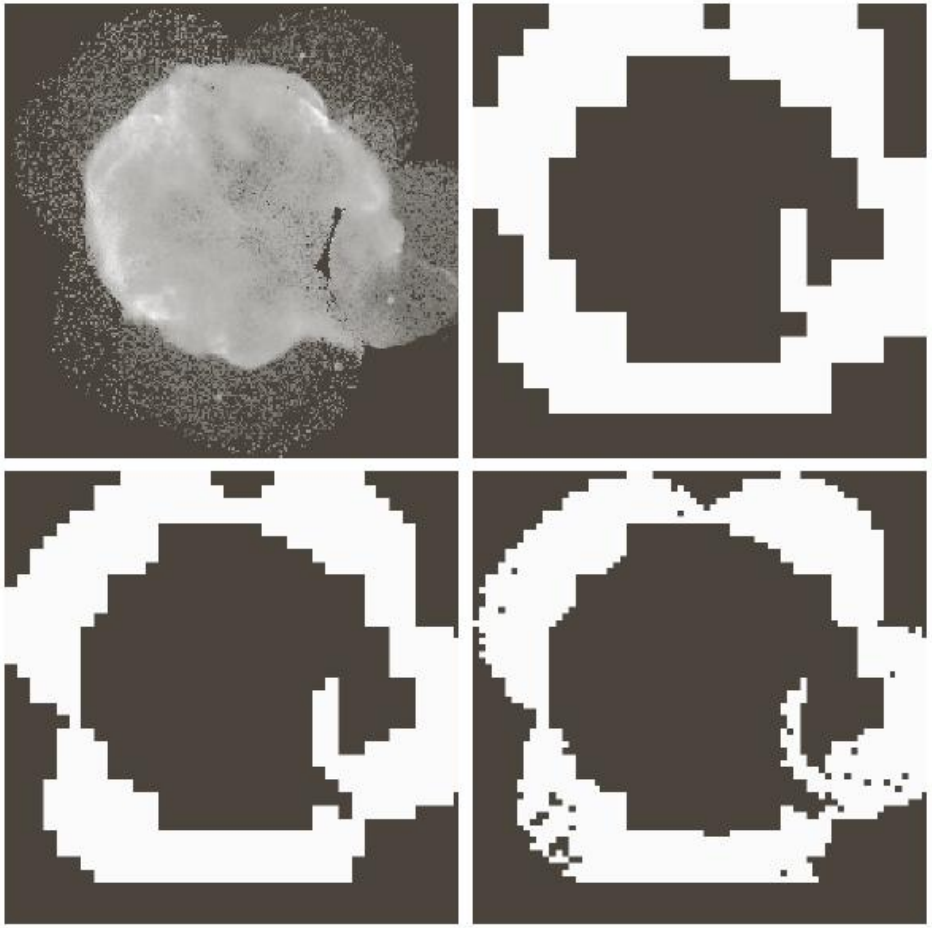
Partición y mezcla (cont.)

- Si solo aplicamos particiones, acabaremos teniendo regiones adyacentes que cumplan (o no) el predicado Q.

Partición y mezcla (cont.)

- Si solo aplicamos particiones, acabaremos teniendo regiones adyacentes que cumplan (o no) el predicado Q.
- Por lo tanto, si $Q(R_j) = Q(R_k)$ para dos regiones adyacentes R_j y R_k, dichas regiones serán mezcladas.

Ejemplo de partición y mezcla



Q =Verdadero si:

- $\sigma > a$
- $0 < m < b$
- m, σ: media y desviación típica de la región.
- a, b: constantes.

Se muestran distintos tamaños mínimos de partición.

Inundación

- Incorpora conceptos de las dos aproximaciones anteriores.

Inundación

- Incorpora conceptos de las dos aproximaciones anteriores.
- Proporciona resultados estables, y genera fronteras cerradas.

Inundación

- Incorpora conceptos de las dos aproximaciones anteriores.
- Proporciona resultados estables, y genera fronteras cerradas.
- Proporciona un marco adecuado para incorporar restricciones basadas en conocimiento previo sobre la imagen.

Inundación

- Incorpora conceptos de las dos aproximaciones anteriores.
- Proporciona resultados estables, y genera fronteras cerradas.
- Proporciona un marco adecuado para incorporar restricciones basadas en conocimiento previo sobre la imagen.

Fundamento de los algoritmos de inundación

Inundación

- Incorpora conceptos de las dos aproximaciones anteriores.
- Proporciona resultados estables, y genera fronteras cerradas.
- Proporciona un marco adecuado para incorporar restricciones basadas en conocimiento previo sobre la imagen.

Fundamento de los algoritmos de inundación

- Se basa en tratar a la imagen como si fuera una superficie tridimensional.

Inundación

- Incorpora conceptos de las dos aproximaciones anteriores.
- Proporciona resultados estables, y genera fronteras cerradas.
- Proporciona un marco adecuado para incorporar restricciones basadas en conocimiento previo sobre la imagen.

Fundamento de los algoritmos de inundación

- Se basa en tratar a la imagen como si fuera una superficie tridimensional.
- La magnitud del gradiente en cada píxel representa la altura del mismo.

Inundación

- Incorpora conceptos de las dos aproximaciones anteriores.
- Proporciona resultados estables, y genera fronteras cerradas.
- Proporciona un marco adecuado para incorporar restricciones basadas en conocimiento previo sobre la imagen.

Fundamento de los algoritmos de inundación

- Se basa en tratar a la imagen como si fuera una superficie tridimensional.
- La magnitud del gradiente en cada píxel representa la altura del mismo.
- Los puntos más altos en esa superficie serán las posibles fronteras.

Representación tridimensional de una imagen

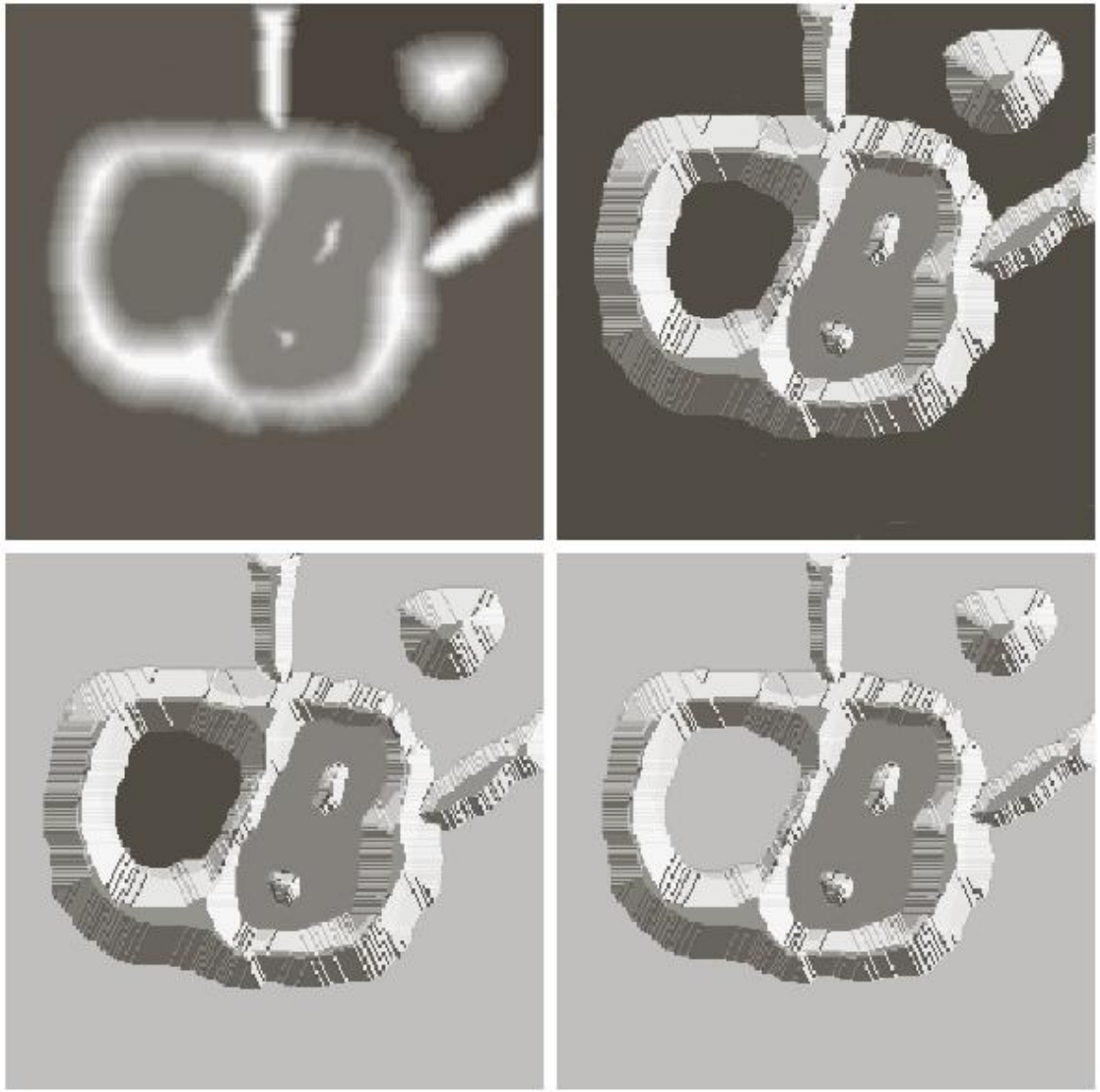


Imagen en tonos de gris, representación tridimensional, y dos niveles de inundación de la misma.

Fundamentos de las técnicas de inundación

- Cada punto de la imagen puede pertenecer a una de las tres siguientes clases:

Fundamentos de las técnicas de inundación

- Cada punto de la imagen puede pertenecer a una de las tres siguientes clases:
  1. Mínimos locales.

Fundamentos de las técnicas de inundación

- Cada punto de la imagen puede pertenecer a una de las tres siguientes clases:
  1. Mínimos locales.
  2. Puntos en los que si se dejara caer una gota de agua, esta iría con certeza hasta un mínimo local concreto.

Fundamentos de las técnicas de inundación

- Cada punto de la imagen puede pertenecer a una de las tres siguientes clases:
  1. Mínimos locales.
  2. Puntos en los que si se dejara caer una gota de agua, esta iría con certeza hasta un mínimo local concreto.
  3. Puntos desde los que una gota podría ir igualmente a más de un mínimo local.

Fundamentos de las técnicas de inundación

- Cada punto de la imagen puede pertenecer a una de las tres siguientes clases:
  1. Mínimos locales.
  2. Puntos en los que si se dejara caer una gota de agua, esta iría con certeza hasta un mínimo local concreto.
  3. Puntos desde los que una gota podría ir igualmente a más de un mínimo local.
- El conjunto de los puntos de la clase 2 que desembocan en un mismo mínimo se denomina *cuenca*.

Fundamentos de las técnicas de inundación

- Cada punto de la imagen puede pertenecer a una de las tres siguientes clases:
  1. Mínimos locales.
  2. Puntos en los que si se dejara caer una gota de agua, esta iría con certeza hasta un mínimo local concreto.
  3. Puntos desde los que una gota podría ir igualmente a más de un mínimo local.
- El conjunto de los puntos de la clase 2 que desembocan en un mismo mínimo se denomina *cuenca*.
- Los puntos (crestas) que cumplen la condición 3 se denominan *líneas divisorias*.

Fundamentos de las técnicas de inundación

- Cada punto de la imagen puede pertenecer a una de las tres siguientes clases:
  1. Mínimos locales.
  2. Puntos en los que si se dejara caer una gota de agua, esta iría con certeza hasta un mínimo local concreto.
  3. Puntos desde los que una gota podría ir igualmente a más de un mínimo local.
- El conjunto de los puntos de la clase 2 que desembocan en un mismo mínimo se denomina *cuenca*.
- Los puntos (crestas) que cumplen la condición 3 se denominan *líneas divisorias*.
- El objetivo de los algoritmos de inundación es encontrar las líneas divisorias.

Algoritmo general de inundación

- Supongamos que la superficie 3D que genera la imagen se va inundando de manera uniforme.

Algoritmo general de inundación

- Supongamos que la superficie 3D que genera la imagen se va inundando de manera uniforme.
- Cuando el agua de dos cuencas distintas está a punto de unirse, se coloca una *presa* para evitar que se mezcle.

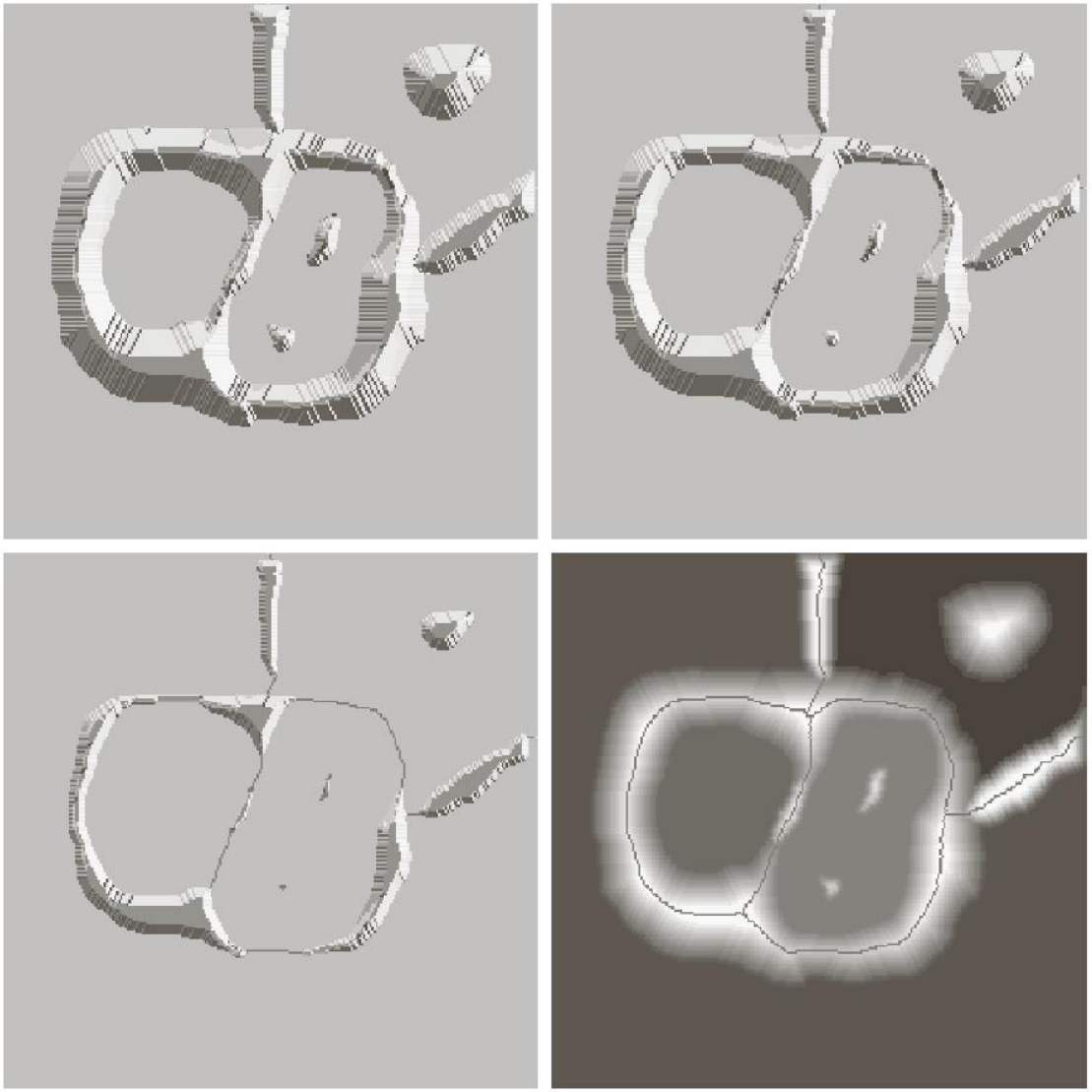
Algoritmo general de inundación

- Supongamos que la superficie 3D que genera la imagen se va inundando de manera uniforme.
- Cuando el agua de dos cuencas distintas está a punto de unirse, se coloca una *presa* para evitar que se mezcle.
- Cuando toda la imagen ha sido inundada, lo único que queda visible son las presas, que corresponderán con las fronteras (conectadas) entre las regiones que buscamos.

Algoritmo general de inundación

- Supongamos que la superficie 3D que genera la imagen se va inundando de manera uniforme.
- Cuando el agua de dos cuencas distintas está a punto de unirse, se coloca una *presa* para evitar que se mezcle.
- Cuando toda la imagen ha sido inundada, lo único que queda visible son las presas, que corresponderán con las fronteras (conectadas) entre las regiones que buscamos.
- La inundación suele aplicarse a la magnitud de gradiente de la imagen.

Ejemplo de inundación



Inundación completa de la imagen, con las *presas* resultantes.

Uso de marcadores

- En ocasiones, produce *sobresegmentación*, debido a irregularidades en el gradiente y al ruido.

Uso de marcadores

- En ocasiones, produce *sobresegmentación*, debido a irregularidades en el gradiente y al ruido.
- Es necesario incorporar conocimiento adicional para limitar el número de regiones que se obtenga.

Uso de marcadores

- En ocasiones, produce *sobresegmentación*, debido a irregularidades en el gradiente y al ruido.
- Es necesario incorporar conocimiento adicional para limitar el número de regiones que se obtenga.

Marcadores

Uso de marcadores

- En ocasiones, produce *sobresegmentación*, debido a irregularidades en el gradiente y al ruido.
- Es necesario incorporar conocimiento adicional para limitar el número de regiones que se obtenga.

Marcadores

- Un *marcador* es un componente conexo de una imagen.

Uso de marcadores

- En ocasiones, produce *sobresegmentación*, debido a irregularidades en el gradiente y al ruido.
- Es necesario incorporar conocimiento adicional para limitar el número de regiones que se obtenga.

Marcadores

- Un *marcador* es un componente conexo de una imagen.
- Hay marcadores internos (asociados a los objetos) y externos (asociados al fondo).

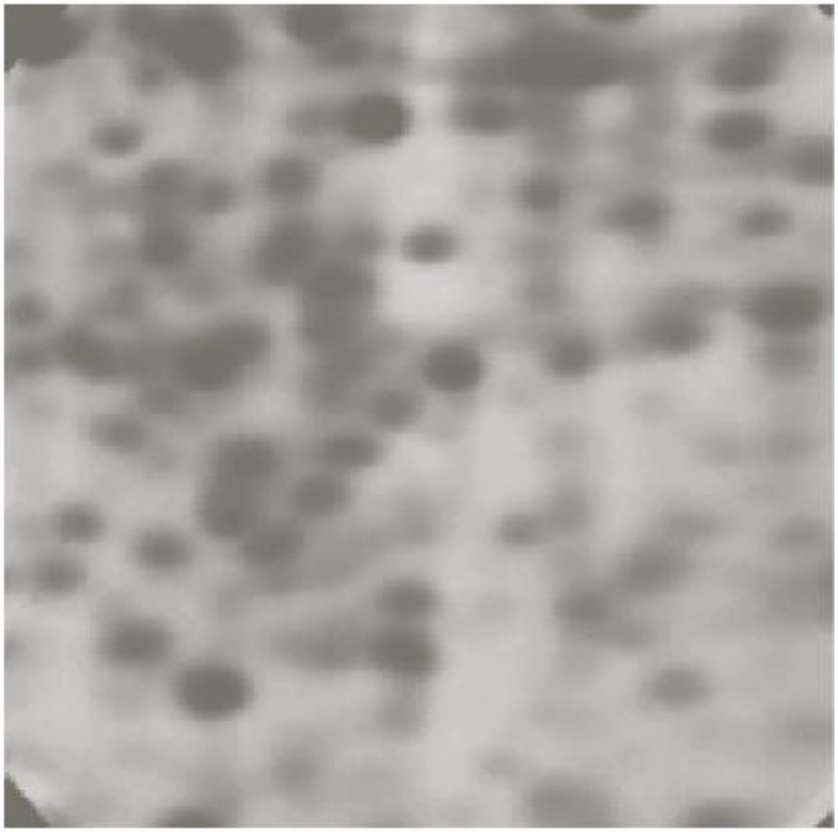
Uso de marcadores

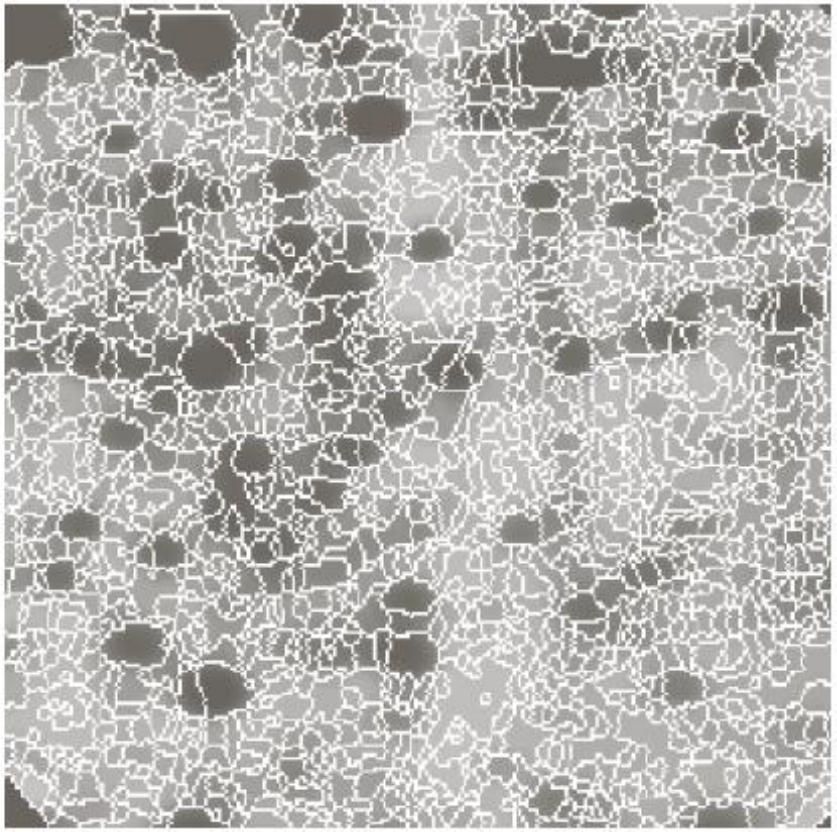
- En ocasiones, produce *sobresegmentación*, debido a irregularidades en el gradiente y al ruido.
- Es necesario incorporar conocimiento adicional para limitar el número de regiones que se obtenga.

Marcadores

- Un *marcador* es un componente conexo de una imagen.
- Hay marcadores internos (asociados a los objetos) y externos (asociados al fondo).
- Se generan en dos pasos: preprocesamiento y definición de un conjunto de criterios que deben satisfacer.

Ejemplo de imagen sobresegmentada





Marcadores internos y externos de la imagen

- *Preprocesamiento*: suavizado. Elimina mínimos locales poco significativos.

Marcadores internos y externos de la imagen

- *Preprocesamiento*: suavizado. Elimina mínimos locales poco significativos.
- *Marcadores internos*: Regiones rodeadas de puntos más altos, que formen una componente cerrada, y que tengan una altura parecida.

Marcadores internos y externos de la imagen

- *Preprocesamiento*: suavizado. Elimina mínimos locales poco significativos.
- *Marcadores internos*: Regiones rodeadas de puntos más altos, que formen una componente cerrada, y que tengan una altura parecida.
- Aplicando la inundación solo a partir de los marcadores internos, las fronteras resultantes serán los *marcadores externos*.

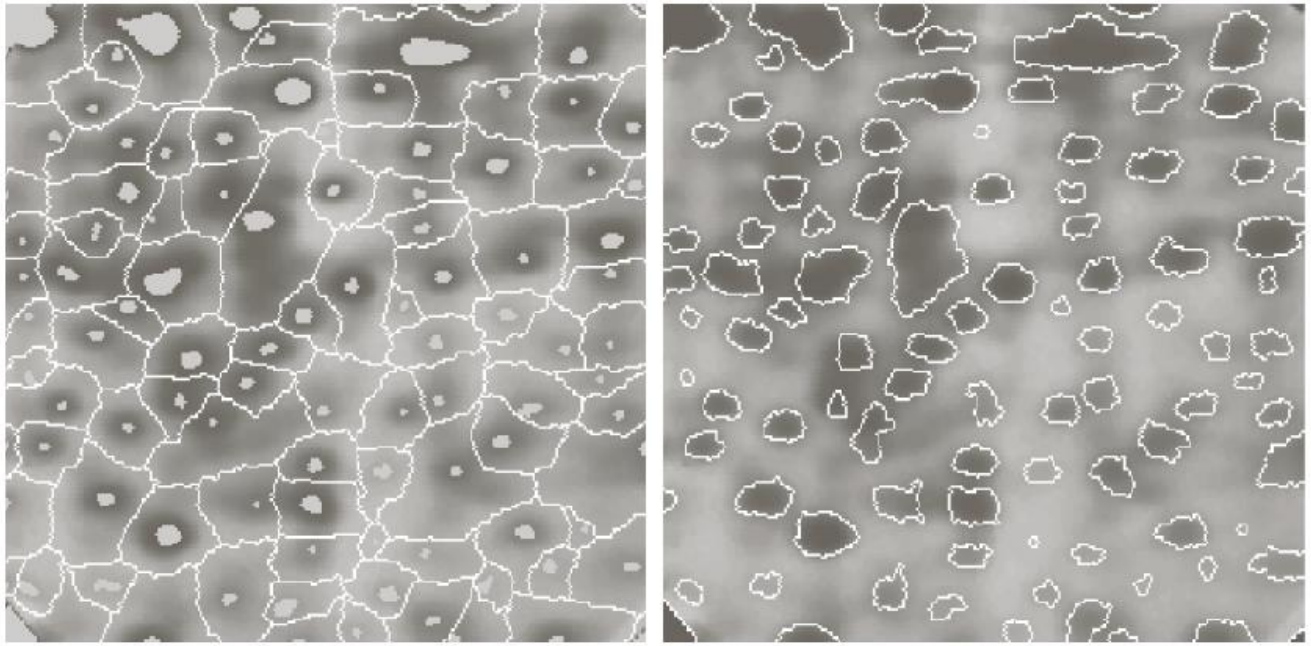
Marcadores internos y externos de la imagen

- *Preprocesamiento*: suavizado. Elimina mínimos locales poco significativos.
- *Marcadores internos*: Regiones rodeadas de puntos más altos, que formen una componente cerrada, y que tengan una altura parecida.
- Aplicando la inundación solo a partir de los marcadores internos, las fronteras resultantes serán los *marcadores externos*.
- Cada región contendrá una zona de interés rodeada por una porción de fondo.

Marcadores internos y externos de la imagen

- *Preprocesamiento*: suavizado. Elimina mínimos locales poco significativos.
- *Marcadores internos*: Regiones rodeadas de puntos más altos, que formen una componente cerrada, y que tengan una altura parecida.
- Aplicando la inundación solo a partir de los marcadores internos, las fronteras resultantes serán los *marcadores externos*.
- Cada región contendrá una zona de interés rodeada por una porción de fondo.
- Estas regiones pueden segmentarse aplicando en cada de ellas, de forma independiente, una nueva inundación.

Resultado de la inundación aplicando marcadores



Izquierda:

- Marcadores internos representados como manchas grises.
- Marcadores externos representados como líneas.

Derecha: Segmentación final.

Índice

1. Segmentación
2. Paradigmas de segmentación
3. Clasificación y reconocimiento
4. Clasificación y reconocimiento mediante *deep learning*

Reconocimiento de patrones en imágenes

Un *patrón* es un conjunto de descriptores organizados de una manera concreta.

Reconocimiento de patrones en imágenes

Un *patrón* es un conjunto de descriptores organizados de una manera concreta.

- Los fenómenos que dan lugar a las imágenes generan patrones característicos.
  - Objetos de la escena, dinámica de los mismos, etc.

Reconocimiento de patrones en imágenes

Un *patrón* es un conjunto de descriptores organizados de una manera concreta.

- Los fenómenos que dan lugar a las imágenes generan patrones característicos.
  - Objetos de la escena, dinámica de los mismos, etc.
- El *reconocimiento* trata de deducir la existencia de determinados elementos en la escena original a partir de los patrones detectados en la(s) imagen(es).

Reconocimiento de patrones en imágenes

Un *patrón* es un conjunto de descriptores organizados de una manera concreta.

- Los fenómenos que dan lugar a las imágenes generan patrones característicos.
  - Objetos de la escena, dinámica de los mismos, etc.
- El *reconocimiento* trata de deducir la existencia de determinados elementos en la escena original a partir de los patrones detectados en la(s) imagen(es).
- Es un problema de decisión estadística que combina información *a priori* con evidencias extraídas de las imágenes.

Reconocimiento de patrones en imágenes

Un *patrón* es un conjunto de descriptores organizados de una manera concreta.

- Los fenómenos que dan lugar a las imágenes generan patrones característicos.
  - Objetos de la escena, dinámica de los mismos, etc.
- El *reconocimiento* trata de deducir la existencia de determinados elementos en la escena original a partir de los patrones detectados en la(s) imagen(es).
- Es un problema de decisión estadística que combina información *a priori* con evidencias extraídas de las imágenes.
- Los patrones suelen representarse mediante vectores, cadenas de caracteres o árboles.

Reconocimiento de patrones en imágenes

Reconocimiento de patrones en imágenes

Reconocimiento de patrones en imágenes

El proceso de reconocimiento

- El sistema de reconocimiento *caracteriza* todos los posibles patrones asociados a un fenómeno u objeto.

El proceso de reconocimiento

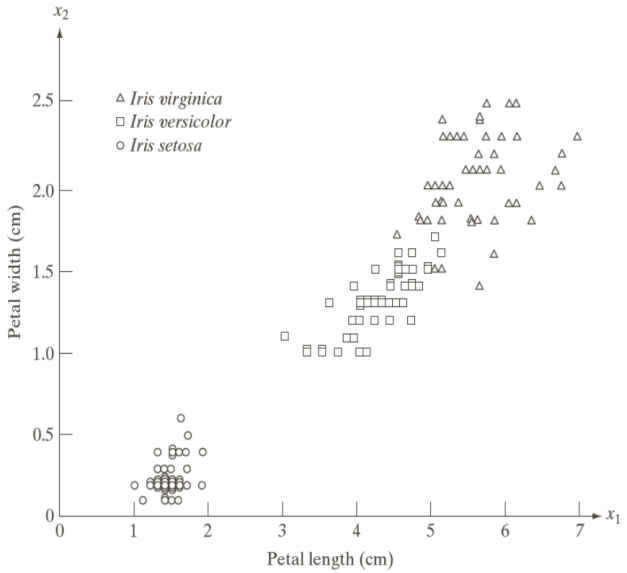
- El sistema de reconocimiento *caracteriza* todos los posibles patrones asociados a un fenómeno u objeto.
- Puede *entrenarse* a partir de ejemplos.

El proceso de reconocimiento

- El sistema de reconocimiento *caracteriza* todos los posibles patrones asociados a un fenómeno u objeto.
- Puede *entrenarse* a partir de ejemplos.
- Dado un patrón observado, decide si el objeto está presente (reconocimiento) y en algunos casos lo asigna a una clase o categoría determinada (clasificación).

El proceso de reconocimiento

- El sistema de reconocimiento *caracteriza* todos los posibles patrones asociados a un fenómeno u objeto.
- Puede *entrenarse* a partir de ejemplos.
- Dado un patrón observado, decide si el objeto está presente (reconocimiento) y en algunos casos lo asigna a una clase o categoría determinada (clasificación).



Esquema general de reconocimiento/clasificación

1. Extraer características de la imagen.

Esquema general de reconocimiento/clasificación

1. Extraer características de la imagen.
2. Obtener una representación de las mismas.

Esquema general de reconocimiento/clasificación

1. Extraer características de la imagen.
2. Obtener una representación de las mismas.
3. Construir un clasificador a partir de ejemplos conocidos.

Esquema general de reconocimiento/clasificación

1. Extraer características de la imagen.
2. Obtener una representación de las mismas.
3. Construir un clasificador a partir de ejemplos conocidos.
4. Clasificar nuevas muestras.

Clasificación bayesiana

- Permite incorporar conocimiento previo al proceso de clasificación.

Clasificación bayesiana

- Permite incorporar conocimiento previo al proceso de clasificación.
- Se basa en el proceso de inferencia probabilística:

$$P(h_i|\mathbf{d}) = \alpha P(\mathbf{d}|h_i)P(h_i)$$

Clasificación bayesiana

- Permite incorporar conocimiento previo al proceso de clasificación.
- Se basa en el proceso de inferencia probabilística:

$$P(h_i|\mathbf{d}) = \alpha P(\mathbf{d}|h_i)P(h_i)$$

h_i: Hipótesis; $\mathbf{d}$: Valor observado; $P(h_i)$: Hipótesis *a priori*; $P(\mathbf{d}|h_i)$: Verosimilitud.

Clasificación bayesiana

- Permite incorporar conocimiento previo al proceso de clasificación.
- Se basa en el proceso de inferencia probabilística:

$$P(h_i|\mathbf{d}) = \alpha P(\mathbf{d}|h_i)P(h_i)$$

h_i: Hipótesis; $\mathbf{d}$: Valor observado; $P(h_i)$: Hipótesis *a priori*; $P(\mathbf{d}|h_i)$: Verosimilitud.

- En algunos casos, calcular la distribución completa para todas las hipótesis es demasiado costoso.

Clasificación bayesiana

- Permite incorporar conocimiento previo al proceso de clasificación.
- Se basa en el proceso de inferencia probabilística:

$$P(h_i|\mathbf{d}) = \alpha P(\mathbf{d}|h_i)P(h_i)$$

h_i: Hipótesis; $\mathbf{d}$: Valor observado; $P(h_i)$: Hipótesis *a priori*; $P(\mathbf{d}|h_i)$: Verosimilitud.

- En algunos casos, calcular la distribución completa para todas las hipótesis es demasiado costoso.
- Hipótesis MAP: nos quedamos con la hipótesis más probable *(máximo a posteriori)*.

Construcción de un reconocedor/clasificador

- Escoger un modelo (red neuronal, red bayesiana, vecinos cercanos, etc.).

Construcción de un reconocedor/clasificador

- Escoger un modelo (red neuronal, red bayesiana, vecinos cercanos, etc.).
- Instanciar el modelo (aprendizaje):
  - Datos de entrenamiento: valores cuya categoría es conocida.
  - Datos de control: valores que se emplean para validar el modelo obtenido con los datos de entrenamiento.

Construcción de un reconocedor/clasificador

- Escoger un modelo (red neuronal, red bayesiana, vecinos cercanos, etc.).
- Instanciar el modelo (aprendizaje):
  - Datos de entrenamiento: valores cuya categoría es conocida.
  - Datos de control: valores que se emplean para validar el modelo obtenido con los datos de entrenamiento.
- Aplicar el modelo a un conjunto de datos de prueba (test) para comprobar la *capacidad de generalización* del reconocedor/clasificador.

Detección de caras mediante características simples en cascada

- Permite detectar la presencia de caras en una imagen.

Detección de caras mediante características simples en cascada

- Permite detectar la presencia de caras en una imagen.
- No distingue una cara de otra (no clasifica).
- Propuesto por Viola y Jones (2001) y mejorado por Lienhart *et al.* en 2002.

Detección de caras mediante características simples en cascada

- Permite detectar la presencia de caras en una imagen.
- No distingue una cara de otra (no clasifica).
- Propuesto por Viola y Jones (2001) y mejorado por Lienhart *et al.* en 2002.
- Emplea características muy simples.

Detección de caras mediante características simples en cascada

- Permite detectar la presencia de caras en una imagen.
- No distingue una cara de otra (no clasifica).
- Propuesto por Viola y Jones (2001) y mejorado por Lienhart *et al.* en 2002.
- Emplea características muy simples.
- Funciona en tiempo real.

Detección de caras mediante características simples en cascada

- Permite detectar la presencia de caras en una imagen.
- No distingue una cara de otra (no clasifica).
- Propuesto por Viola y Jones (2001) y mejorado por Lienhart *et al.* en 2002.
- Emplea características muy simples.
- Funciona en tiempo real.
- Usa una serie de clasificadores *débiles* en cascada.

Características de Haar

- Consisten en tomar un área de la imagen, subdividirla en dos o más regiones, y sumar (o restar) las sumas de los píxeles en cada una de ellas.

Características de Haar

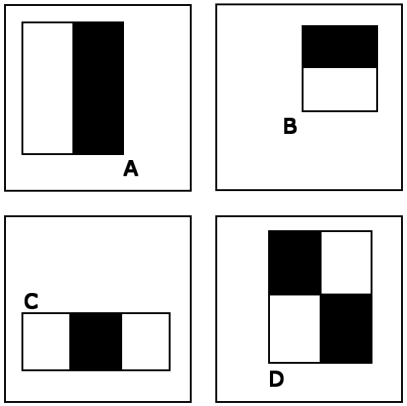
- Consisten en tomar un área de la imagen, subdividirla en dos o más regiones, y sumar (o restar) las sumas de los píxeles en cada una de ellas.
- Fijando un umbral, se puede detectar la presencia de determinados patrones en el área de la imagen.

Características de Haar

- Consisten en tomar un área de la imagen, subdividirla en dos o más regiones, y sumar (o restar) las sumas de los píxeles en cada una de ellas.
- Fijando un umbral, se puede detectar la presencia de determinados patrones en el área de la imagen.
- Dada una ventana, es posible evaluar distintas caraterísticas en distintas posiciones de la cara.

Características de Haar

- Consisten en tomar un área de la imagen, subdividirla en dos o más regiones, y sumar (o restar) las sumas de los píxeles en cada una de ellas.
- Fijando un umbral, se puede detectar la presencia de determinados patrones en el área de la imagen.
- Dada una ventana, es posible evaluar distintas caraterísticas en distintas posiciones de la cara.



Características usadas en la propuesta de Viola y Jones. Las zonas claras se suman, y las oscuras se restan.

Detectores *débiles* en cascada

- Evaluar una única característica produce tasas de error muy grandes.

Detectores *débiles* en cascada

- Evaluar una única característica produce tasas de error muy grandes.
- Es posible fijar el umbral para que los falsos negativos sean muy próximos a cero, aunque los falsos positivos sean muchos (30 o 40 %).

Detectores *débiles* en cascada

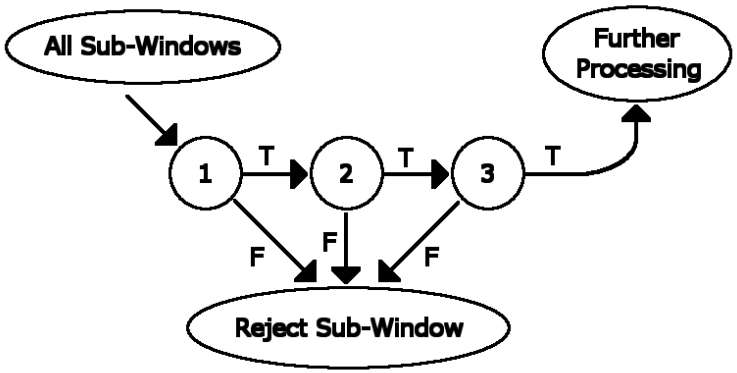
- Evaluar una única característica produce tasas de error muy grandes.
- Es posible fijar el umbral para que los falsos negativos sean muy próximos a cero, aunque los falsos positivos sean muchos (30 o 40 %).
- Colocando muchos detectores de este tipo en cascada se reducen drásticamente los falsos positivos.

Detectores *débiles* en cascada

- Evaluar una única característica produce tasas de error muy grandes.
- Es posible fijar el umbral para que los falsos negativos sean muy próximos a cero, aunque los falsos positivos sean muchos (30 o 40 %).
- Colocando muchos detectores de este tipo en cascada se reducen drásticamente los falsos positivos.
- Evaluando primero las características más discriminantes, se obtiene un sistema mucho más rápido.

Detectores *débiles* en cascada

- Evaluar una única característica produce tasas de error muy grandes.
- Es posible fijar el umbral para que los falsos negativos sean muy próximos a cero, aunque los falsos positivos sean muchos (30 o 40 %).
- Colocando muchos detectores de este tipo en cascada se reducen drásticamente los falsos positivos.
- Evaluando primero las características más discriminantes, se obtiene un sistema mucho más rápido.



Solo evaluamos la siguiente característica si todas las anteriores han dado positivo.

Evaluación rápida de las características de Haar

- Se parte de la llamada *imagen integral*, $I(x, y)$, en la que cada punto es la suma de todos los píxeles que quedan por encima y a la izquierda.

Evaluación rápida de las características de Haar

- Se parte de la llamada *imagen integral*, $I(x, y)$, en la que cada punto es la suma de todos los píxeles que quedan por encima y a la izquierda.
- Puede calcularse en una única pasada:

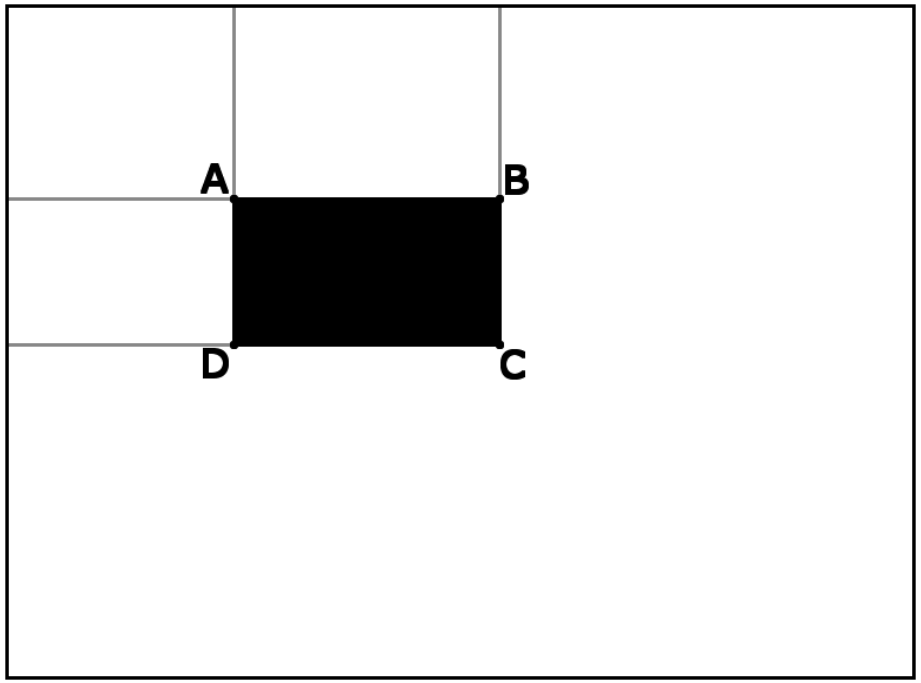
$$I(x, y) = i(x, y) + I(x - 1, y) + I(x, y - 1) - I(x - 1, y - 1)$$

Siendo $i(x, y)$ un píxel de la imagen.

Evaluación rápida de las características de Haar

- La suma de los píxeles de una región con esquinas A, B, C, D se calcula:

$$I(C) + I(A) - I(B) - I(D)$$



Detector de Viola y Jones

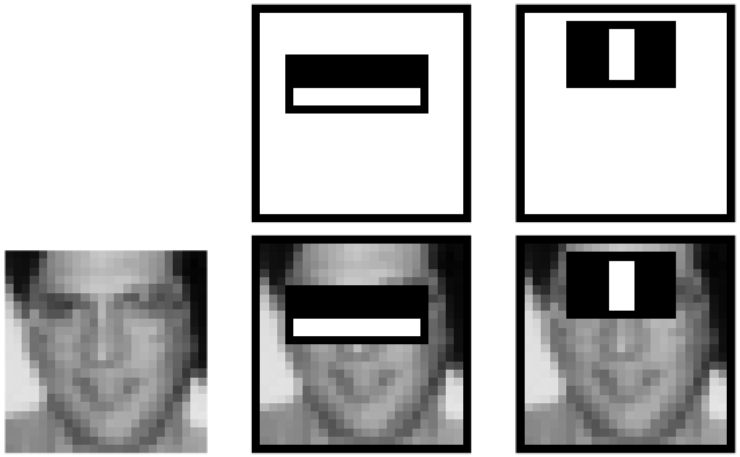
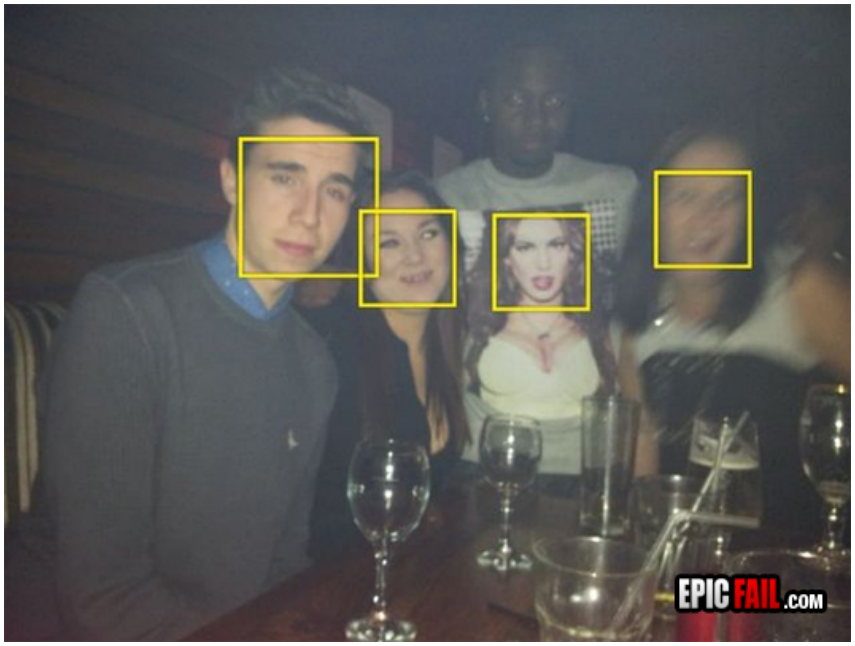


Figure 5: Example of frontal upright face images used for training.

El detector no siempre funciona...



Reconocimiento de objetos en general

- Consiste en detectar la presencia de un objeto en la imagen.

Reconocimiento de objetos en general

- Consiste en detectar la presencia de un objeto en la imagen.
- Pueden calcularse parámetros del mismo (posición en la imagen, posición 3D en la escena, etc.).

Reconocimiento de objetos en general

- Consiste en detectar la presencia de un objeto en la imagen.
- Pueden calcularse parámetros del mismo (posición en la imagen, posición 3D en la escena, etc.).
- Propiedades deseables del detector:

Reconocimiento de objetos en general

- Consiste en detectar la presencia de un objeto en la imagen.
- Pueden calcularse parámetros del mismo (posición en la imagen, posición 3D en la escena, etc.).
- Propiedades deseables del detector:
  - Invariante a cambios de posición y orientación.

Reconocimiento de objetos en general

- Consiste en detectar la presencia de un objeto en la imagen.
- Pueden calcularse parámetros del mismo (posición en la imagen, posición 3D en la escena, etc.).
- Propiedades deseables del detector:
  - Invariante a cambios de posición y orientación.
  - Resistente a oclusiones

Reconocimiento de objetos en general

- Consiste en detectar la presencia de un objeto en la imagen.
- Pueden calcularse parámetros del mismo (posición en la imagen, posición 3D en la escena, etc.).
- Propiedades deseables del detector:
  - Invariante a cambios de posición y orientación.
  - Resistente a oclusiones
- Tipos de objetos:

Reconocimiento de objetos en general

- Consiste en detectar la presencia de un objeto en la imagen.
- Pueden calcularse parámetros del mismo (posición en la imagen, posición 3D en la escena, etc.).
- Propiedades deseables del detector:
  - Invariante a cambios de posición y orientación.
  - Resistente a oclusiones
- Tipos de objetos:
  - Rígidos, deformables, articulados, etc.

Reconocedores basados en apariencia

- Usan una representación completa del objeto (modelo).

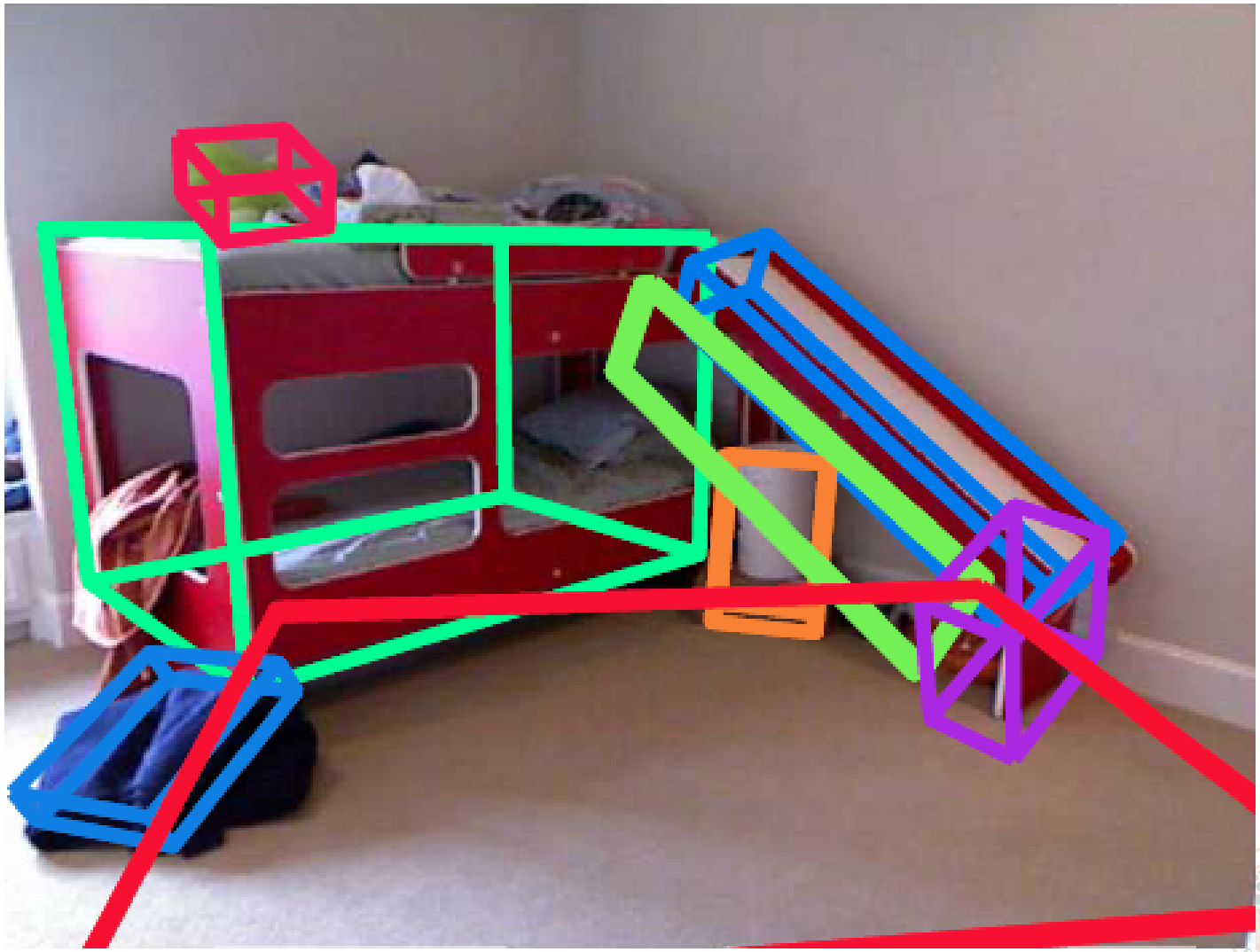
Reconocedores basados en apariencia

- Usan una representación completa del objeto (modelo).
- El modelo contiene parámetros que generan las posibles apariencias del objeto.
- Intentan alinear el modelo con la imagen *(matching)*, recuperando el valor de los parámetros que da lugar a la apariencia del objeto en la imagen.

Reconocedores basados en apariencia

- Usan una representación completa del objeto (modelo).
- El modelo contiene parámetros que generan las posibles apariencias del objeto.
- Intentan alinear el modelo con la imagen *(matching)*, recuperando el valor de los parámetros que da lugar a la apariencia del objeto en la imagen.
- Hay que tener en cuenta cambios de perspectiva, color, tamaño, iluminación.

Ejemplo de reconocimiento basado en apariencia



Reconocedores basados en características

- A partir de una serie de ejemplos, se determinan *características* propias del objeto en distintas condiciones.

Reconocedores basados en características

- A partir de una serie de ejemplos, se determinan *características* propias del objeto en distintas condiciones.
- Hay que buscar puntos *interesantes*: esquinas, *blobs*, etc.

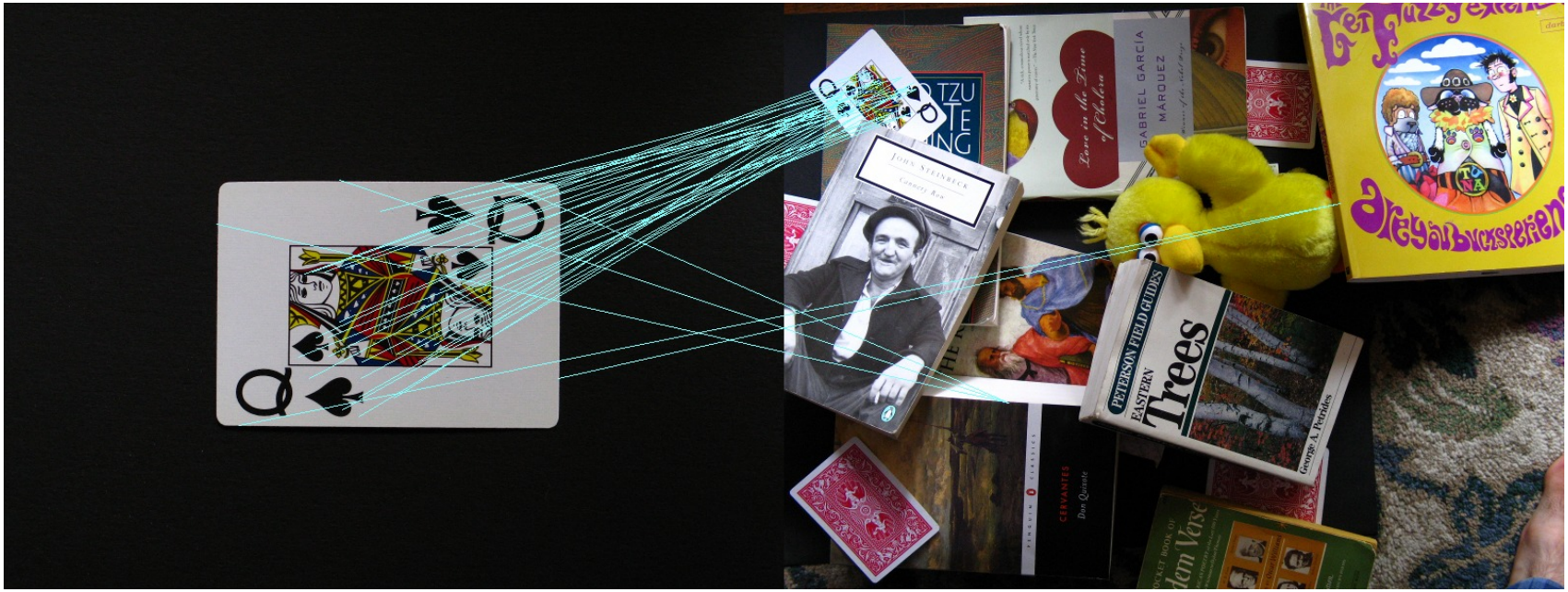
Reconocedores basados en características

- A partir de una serie de ejemplos, se determinan *características* propias del objeto en distintas condiciones.
- Hay que buscar puntos *interesantes*: esquinas, *blobs*, etc.
- Se calcula algún descriptor con determinadas invarianzas (SIFT, SURF, etc.) en cada una de ellas.

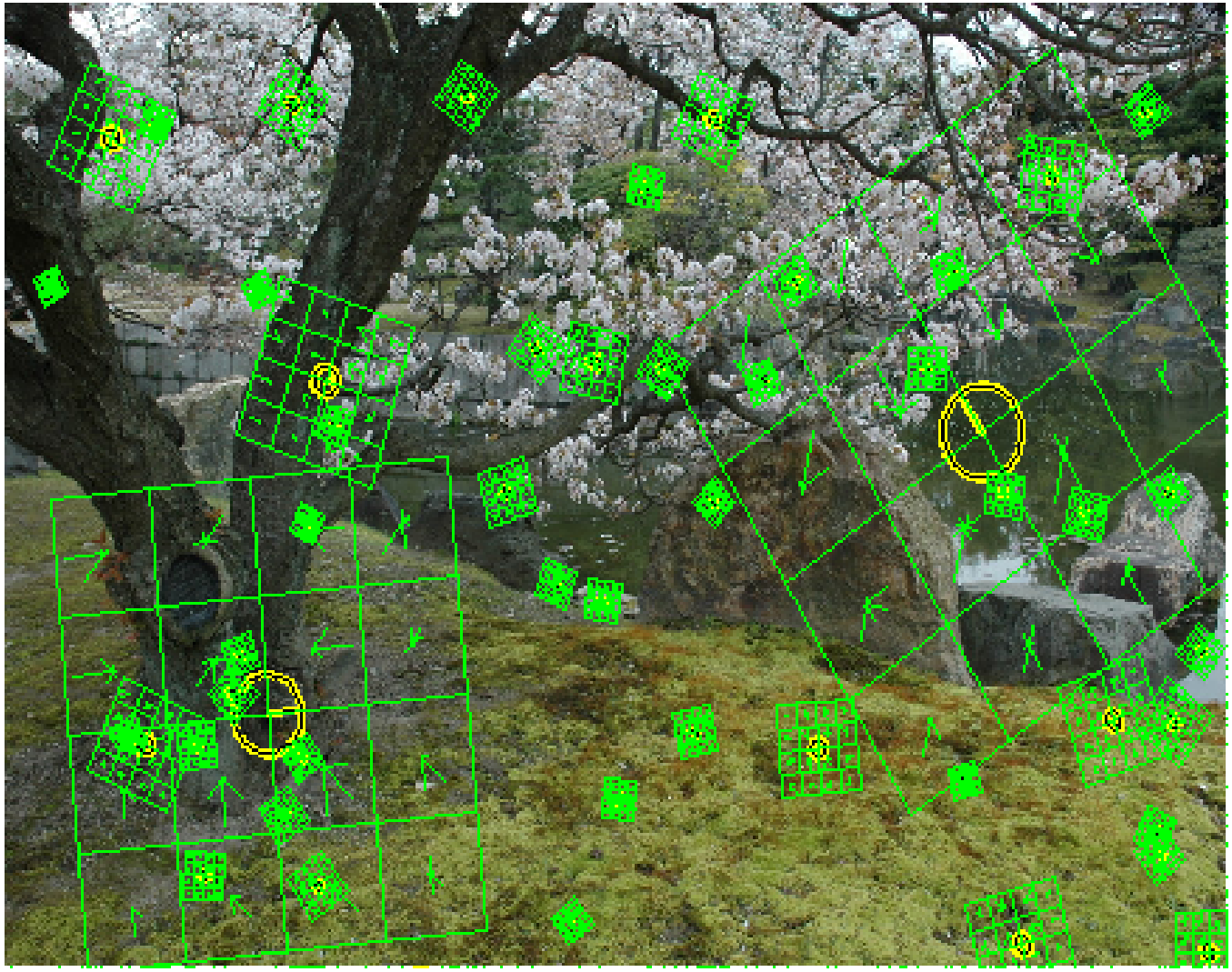
Reconocedores basados en características

- A partir de una serie de ejemplos, se determinan *características* propias del objeto en distintas condiciones.
- Hay que buscar puntos *interesantes*: esquinas, *blobs*, etc.
- Se calcula algún descriptor con determinadas invarianzas (SIFT, SURF, etc.) en cada una de ellas.
- En la imagen a clasificar, se considera que está el objeto si aparecen suficientes características en posiciones coherentes.

Ejemplos de reconocimiento basado en características



Ejemplos de reconocimiento basado en características

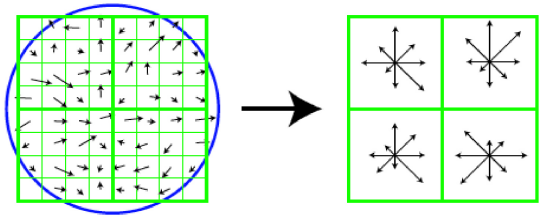


Descriptor SIFT (Lowe, 1999)

- Proporciona un algoritmo para detectar *blobs* en diferentes escalas.

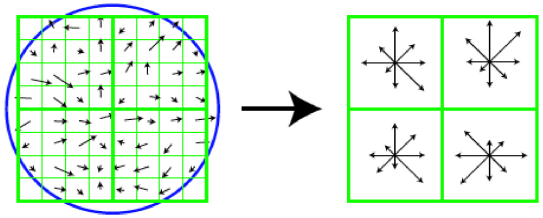
Descriptor SIFT (Lowe, 1999)

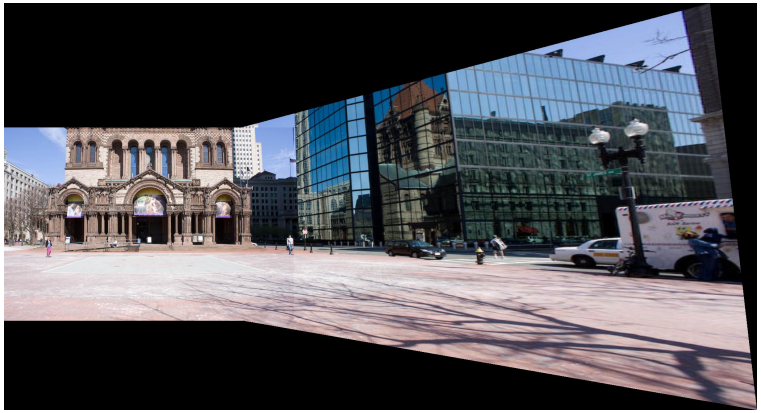
- Proporciona un algoritmo para detectar *blobs* en diferentes escalas.
- El descriptor se basa en histogramas locales de orientación de gradiente.



Descriptor SIFT (Lowe, 1999)

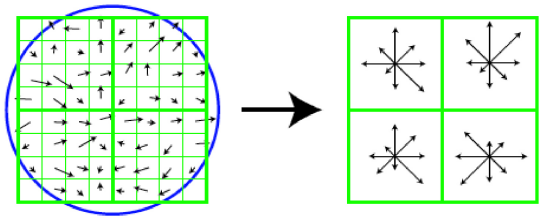
- Proporciona un algoritmo para detectar *blobs* en diferentes escalas.
- El descriptor se basa en histogramas locales de orientación de gradiente.
- No solo permite detectar objetos, sino alinear (registrar) imágenes.

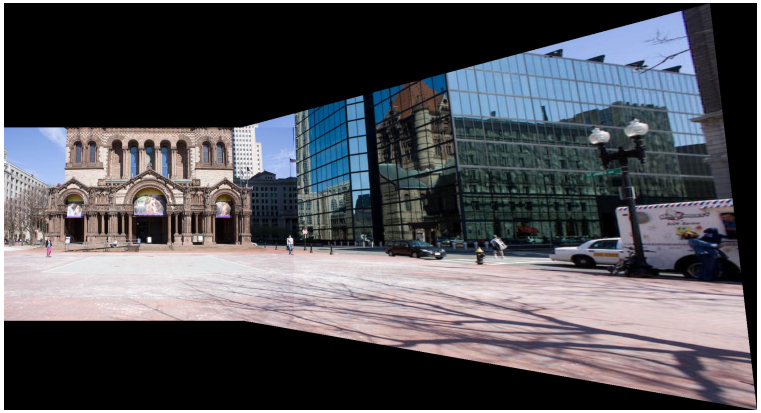


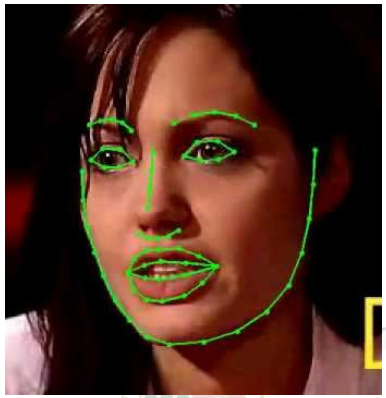


Descriptor SIFT (Lowe, 1999)

- Proporciona un algoritmo para detectar *blobs* en diferentes escalas.
- El descriptor se basa en histogramas locales de orientación de gradiente.
- No solo permite detectar objetos, sino alinear (registrar) imágenes.
- También se usa para localizar puntos de referencia *(landmarks)*, por ejemplo en reconocimiento de gestos y caras.







Invarianzas de SIFT

- Invariante a traslación, escalado y rotación.

Invarianzas de SIFT

- Invariante a traslación, escalado y rotación.
- Parcialmente invariante a cambios de iluminación.

Invarianzas de SIFT

- Invariante a traslación, escalado y rotación.
- Parcialmente invariante a cambios de iluminación.
- *Robusto* frente distorsiones geométricas locales.

Invarianzas de SIFT

- Invariante a traslación, escalado y rotación.
- Parcialmente invariante a cambios de iluminación.
- *Robusto* frente distorsiones geométricas locales.
- Propiedades similares a las neuronas de la corteza temporal inferior, empleada por los primates para reconocer objetos.

Espacio de escalas y detección de *blobs*

Espacio de escalas

1. Un filtro de Gaussiana para un valor determinado de σ (escala) tiene la forma:

$$g(x, y; \sigma) = \frac{1}{2\pi\sigma} e^{-(x^2+y^2)/2\sigma}$$

Espacio de escalas y detección de *blobs*

Espacio de escalas

1. Un filtro de Gaussiana para un valor determinado de σ (escala) tiene la forma:

$$g(x, y; \sigma) = \frac{1}{2\pi\sigma} e^{-(x^2+y^2)/2\sigma}$$

2. Se puede generar un volumen de imágenes de la forma:

$$L(\cdot, \cdot; \sigma) = g(\cdot, \cdot; \sigma) * f(\cdot, \cdot)$$

Espacio de escalas y detección de *blobs*

Espacio de escalas

1. Un filtro de Gaussiana para un valor determinado de σ (escala) tiene la forma:

$$g(x, y; \sigma) = \frac{1}{2\pi\sigma} e^{-(x^2+y^2)/2\sigma}$$

2. Se puede generar un volumen de imágenes de la forma:

$$L(\cdot, \cdot; \sigma) = g(\cdot, \cdot; \sigma) * f(\cdot, \cdot)$$

   - Este volumen se conoce como *espacio de escala*.

Espacio de escalas y detección de *blobs*

Espacio de escalas

1. Un filtro de Gaussiana para un valor determinado de σ (escala) tiene la forma:

$$g(x, y; \sigma) = \frac{1}{2\pi\sigma} e^{-(x^2+y^2)/2\sigma}$$

2. Se puede generar un volumen de imágenes de la forma:

$$L(\cdot, \cdot; \sigma) = g(\cdot, \cdot; \sigma) * f(\cdot, \cdot)$$

- Este volumen se conoce como *espacio de escala*.
- Sus tres dimensiones son (x, y, σ).

Espacio de escalas y detección de *blobs*

Espacio de escalas

1. Un filtro de Gaussiana para un valor determinado de σ (escala) tiene la forma:

$$g(x, y; \sigma) = \frac{1}{2\pi\sigma} e^{-(x^2+y^2)/2\sigma}$$

2. Se puede generar un volumen de imágenes de la forma:

$$L(\cdot, \cdot; \sigma) = g(\cdot, \cdot; \sigma) * f(\cdot, \cdot)$$

- Este volumen se conoce como *espacio de escala*.
- Sus tres dimensiones son (x, y, σ).
- En el origen ($\sigma = 0$), está la imagen original.

Espacio de escalas y detección de *blobs*

Espacio de escalas y detección de *blobs*

1. Las segundas derivadas del espacio de escala con respecto a x e y presentan extremos locales (en las tres dimensiones) donde haya manchas *(blobs)* en la imagen.

Espacio de escalas y detección de *blobs*

1. Las segundas derivadas del espacio de escala con respecto a x e y presentan extremos locales (en las tres dimensiones) donde haya manchas *(blobs)* en la imagen.
   - La posición (x, y) es el centro de la mancha.

Espacio de escalas y detección de *blobs*

1. Las segundas derivadas del espacio de escala con respecto a x e y presentan extremos locales (en las tres dimensiones) donde haya manchas *(blobs)* en la imagen.
   - La posición (x, y) es el centro de la mancha.
   - σ es la escala (tamaño) de la misma.

Espacio de escalas y detección de *blobs*

1. Las segundas derivadas del espacio de escala con respecto a x e y presentan extremos locales (en las tres dimensiones) donde haya manchas *(blobs)* en la imagen.
   - La posición (x, y) es el centro de la mancha.
   - σ es la escala (tamaño) de la misma.
2. La segunda derivada se puede aproximar mediante diferencias de gaussianas (DoG), que consiste en calcular la diferencia entre dos imágenes convolucionadas con valores diferentes de σ:

$$\nabla^2 L(x, y; \sigma) \approx \frac{\sigma}{\Delta\sigma}(L(x, y; \sigma + \Delta\sigma) - L(x, y; \sigma))$$

Índice

1. Segmentación
2. Paradigmas de segmentación
3. Clasificación y reconocimiento
4. Clasificación y reconocimiento mediante *deep learning*

Clasificación y reconocimiento

- Cuando tenemos datos de entrada (imágenes, o cualquier otra cosa) representados mediante secuencias de símbolos, nuestro objetivo es que el computador sea capaz de decirnos si pertenecen a alguna categoría determinada, o si contienen determinado fenómeno.

Clasificación y reconocimiento

- Cuando tenemos datos de entrada (imágenes, o cualquier otra cosa) representados mediante secuencias de símbolos, nuestro objetivo es que el computador sea capaz de decirnos si pertenecen a alguna categoría determinada, o si contienen determinado fenómeno.
- La *clasificación* hace referencia a la separación en categorías de los datos, mientras que el *reconocimiento* es un concepto más amplio, que puede proporcionar otros tipos de respuesta:

Clasificación y reconocimiento

- Cuando tenemos datos de entrada (imágenes, o cualquier otra cosa) representados mediante secuencias de símbolos, nuestro objetivo es que el computador sea capaz de decirnos si pertenecen a alguna categoría determinada, o si contienen determinado fenómeno.
- La *clasificación* hace referencia a la separación en categorías de los datos, mientras que el *reconocimiento* es un concepto más amplio, que puede proporcionar otros tipos de respuesta:
  - Regresión: Proporciona como salidas números reales en lugar de categorías. Ejemplo: la posición y tamaño de un objeto en una imagen.

Clasificación y reconocimiento

- Cuando tenemos datos de entrada (imágenes, o cualquier otra cosa) representados mediante secuencias de símbolos, nuestro objetivo es que el computador sea capaz de decirnos si pertenecen a alguna categoría determinada, o si contienen determinado fenómeno.
- La *clasificación* hace referencia a la separación en categorías de los datos, mientras que el *reconocimiento* es un concepto más amplio, que puede proporcionar otros tipos de respuesta:
  - Regresión: Proporciona como salidas números reales en lugar de categorías. Ejemplo: la posición y tamaño de un objeto en una imagen.
  - Etiquetado de secuencias: Por ejemplo, etiquetado de palabras en un fragmento de voz humana.

Clasificación y reconocimiento

- Cuando tenemos datos de entrada (imágenes, o cualquier otra cosa) representados mediante secuencias de símbolos, nuestro objetivo es que el computador sea capaz de decirnos si pertenecen a alguna categoría determinada, o si contienen determinado fenómeno.
- La *clasificación* hace referencia a la separación en categorías de los datos, mientras que el *reconocimiento* es un concepto más amplio, que puede proporcionar otros tipos de respuesta:
  - Regresión: Proporciona como salidas números reales en lugar de categorías. Ejemplo: la posición y tamaño de un objeto en una imagen.
  - Etiquetado de secuencias: Por ejemplo, etiquetado de palabras en un fragmento de voz humana.
  - Análisis sintácticos: Análisis de la estructura de una secuencia.

Clasificación y reconocimiento

- Cuando tenemos datos de entrada (imágenes, o cualquier otra cosa) representados mediante secuencias de símbolos, nuestro objetivo es que el computador sea capaz de decirnos si pertenecen a alguna categoría determinada, o si contienen determinado fenómeno.
- La *clasificación* hace referencia a la separación en categorías de los datos, mientras que el *reconocimiento* es un concepto más amplio, que puede proporcionar otros tipos de respuesta:
  - Regresión: Proporciona como salidas números reales en lugar de categorías. Ejemplo: la posición y tamaño de un objeto en una imagen.
  - Etiquetado de secuencias: Por ejemplo, etiquetado de palabras en un fragmento de voz humana.
  - Análisis sintácticos: Análisis de la estructura de una secuencia.
- En Visión Artificial se usan sobre todo clasificaciones (¿aparece el objeto X en la imagen?) y regresiones (dónde y con qué tamaño).

Redes neuronales

- Son conjuntos conectados de neuronas, organizadas en capas.

Redes neuronales

- Son conjuntos conectados de neuronas, organizadas en capas.
- Una neurona tiene las siguientes partes:

Redes neuronales

- Son conjuntos conectados de neuronas, organizadas en capas.
- Una neurona tiene las siguientes partes:
  - Un conjunto de entradas x_i, con *pesos* w_i asociados.

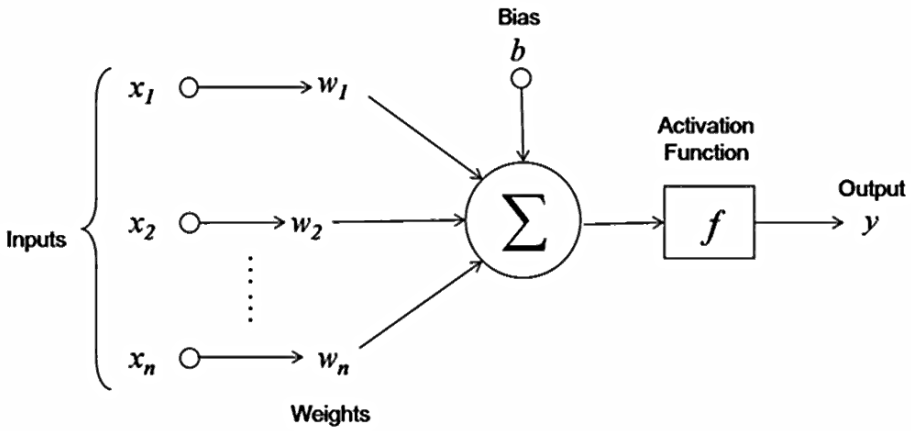
Redes neuronales

- Son conjuntos conectados de neuronas, organizadas en capas.
- Una neurona tiene las siguientes partes:
  - Un conjunto de entradas x_i, con *pesos* w_i asociados.
  - Una función de activación $f()$.

Redes neuronales

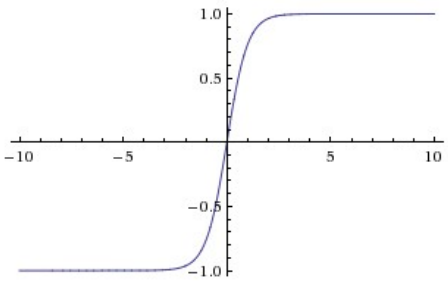
- Son conjuntos conectados de neuronas, organizadas en capas.
- Una neurona tiene las siguientes partes:
  - Un conjunto de entradas x_i, con *pesos* w_i asociados.
  - Una función de activación $f()$.
- La salida de la neurona es:

$$f\left(\sum x_i \cdot w_i + b\right)$$

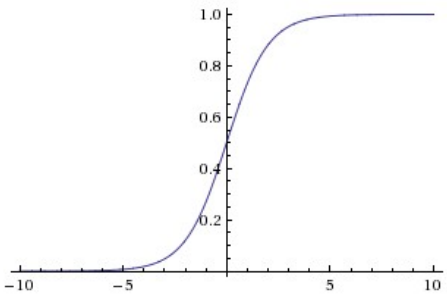


La función de activación $f()$

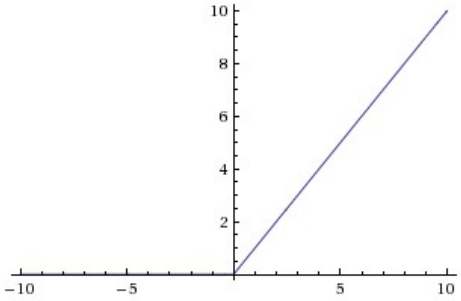
- Proporciona la salida de la neurona.



Tanh



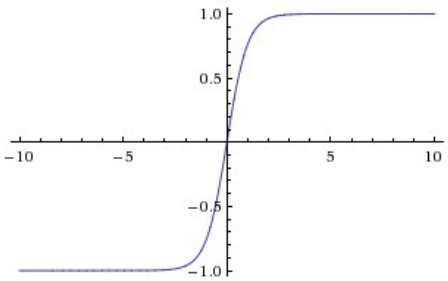
Sigmoide



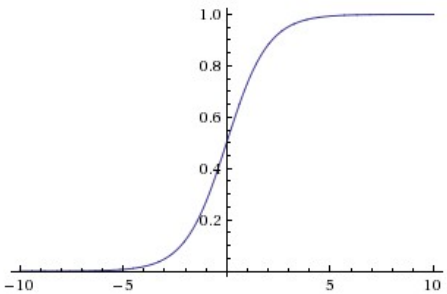
ReLU

La función de activación $f()$

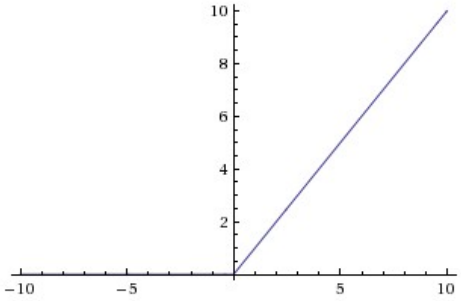
- Proporciona la salida de la neurona.
- Suele ser *no lineal* y monótona.



Tanh



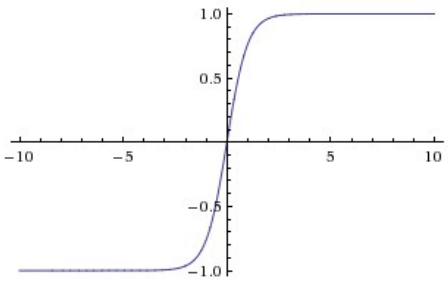
Sigmoide



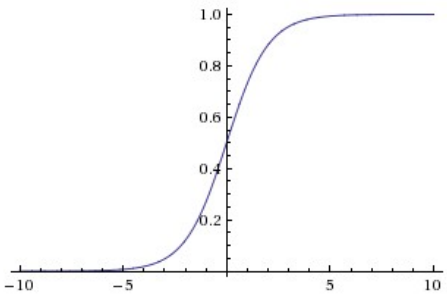
ReLU

La función de activación $f()$

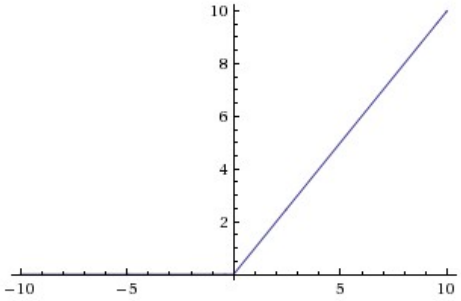
- Proporciona la salida de la neurona.
- Suele ser *no lineal* y monótona.
- Debe ser derivable.



Tanh



Sigmoide



ReLU

La neurona como elemento clasificador

- Si consideramos el hiperespacio dado por los valores de entrada x_i, los puntos del mismo que presentan un valor concreto para la función de activación $f()$ forman un hiperplano.

La neurona como elemento clasificador

- Si consideramos el hiperespacio dado por los valores de entrada x_i, los puntos del mismo que presentan un valor concreto para la función de activación $f()$ forman un hiperplano.
- La neurona divide el hiperespacio en dos mitades.

La neurona como elemento clasificador

- Si consideramos el hiperespacio dado por los valores de entrada x_i, los puntos del mismo que presentan un valor concreto para la función de activación $f()$ forman un hiperplano.
- La neurona divide el hiperespacio en dos mitades.
- El hiperplano viene dado por los pesos w_i y por el sesgo *(bias)* b.

La neurona como elemento clasificador

- Si consideramos el hiperespacio dado por los valores de entrada x_i, los puntos del mismo que presentan un valor concreto para la función de activación $f()$ forman un hiperplano.
- La neurona divide el hiperespacio en dos mitades.
- El hiperplano viene dado por los pesos w_i y por el sesgo *(bias)* b.
- El objetivo es *separar* el hiperespacio en zonas correspondientes a las categorías.

La neurona como elemento clasificador

- Si consideramos el hiperespacio dado por los valores de entrada x_i, los puntos del mismo que presentan un valor concreto para la función de activación $f()$ forman un hiperplano.
- La neurona divide el hiperespacio en dos mitades.
- El hiperplano viene dado por los pesos w_i y por el sesgo *(bias)* b.
- El objetivo es *separar* el hiperespacio en zonas correspondientes a las categorías.
  - Dos clases: una neurona.

La neurona como elemento clasificador

- Si consideramos el hiperespacio dado por los valores de entrada x_i, los puntos del mismo que presentan un valor concreto para la función de activación $f()$ forman un hiperplano.
- La neurona divide el hiperespacio en dos mitades.
- El hiperplano viene dado por los pesos w_i y por el sesgo *(bias)* b.
- El objetivo es *separar* el hiperespacio en zonas correspondientes a las categorías.
  - Dos clases: una neurona.
  - Múltiples clases: una neurona por clase, y nos quedamos con la clase de la neurona que dé máxima respuesta *(softmax)*.

La neurona como elemento clasificador

- Si consideramos el hiperespacio dado por los valores de entrada x_i, los puntos del mismo que presentan un valor concreto para la función de activación $f()$ forman un hiperplano.
- La neurona divide el hiperespacio en dos mitades.
- El hiperplano viene dado por los pesos w_i y por el sesgo *(bias)* b.
- El objetivo es *separar* el hiperespacio en zonas correspondientes a las categorías.
  - Dos clases: una neurona.
  - Múltiples clases: una neurona por clase, y nos quedamos con la clase de la neurona que dé máxima respuesta *(softmax)*.

`http://playground.tensorflow.org/`

Clasificación mediante redes neuronales

- En representaciones complejas, con vectores grandes, es difícil que haya separaciones claras.

Clasificación mediante redes neuronales

- En representaciones complejas, con vectores grandes, es difícil que haya separaciones claras.
- Introduciendo capas de neuronas, se obtienen transformaciones del espacio vectorial de entrada que hacen las clases más *separables*.

Clasificación mediante redes neuronales

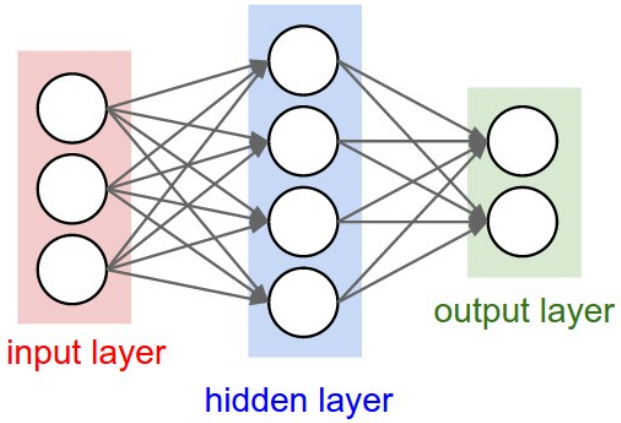
- En representaciones complejas, con vectores grandes, es difícil que haya separaciones claras.
- Introduciendo capas de neuronas, se obtienen transformaciones del espacio vectorial de entrada que hacen las clases más *separables*.
- Las salidas de las neuronas en una capa forman un hiperespacio transformado a partir del original, que puede tener un número diferente de dimensiones.

Clasificación mediante redes neuronales

- En representaciones complejas, con vectores grandes, es difícil que haya separaciones claras.
- Introduciendo capas de neuronas, se obtienen transformaciones del espacio vectorial de entrada que hacen las clases más *separables*.
- Las salidas de las neuronas en una capa forman un hiperespacio transformado a partir del original, que puede tener un número diferente de dimensiones.
- El aprendizaje profundo *(deep learning)* trata del uso de redes neuronales de múltiples capas para resolver problemas complejos.

Clasificación mediante redes neuronales

- En representaciones complejas, con vectores grandes, es difícil que haya separaciones claras.
- Introduciendo capas de neuronas, se obtienen transformaciones del espacio vectorial de entrada que hacen las clases más *separables*.
- Las salidas de las neuronas en una capa forman un hiperespacio transformado a partir del original, que puede tener un número diferente de dimensiones.
- El aprendizaje profundo *(deep learning)* trata del uso de redes neuronales de múltiples capas para resolver problemas complejos.



Complejidad y generalización

- Cuanto más complejo es un sistema de aprendizaje, más probable es que *sobreaprenda* (memorize cada uno de los ejemplos de entrenamiento de forma individual), y que sea incapaz de generalizar.

Complejidad y generalización

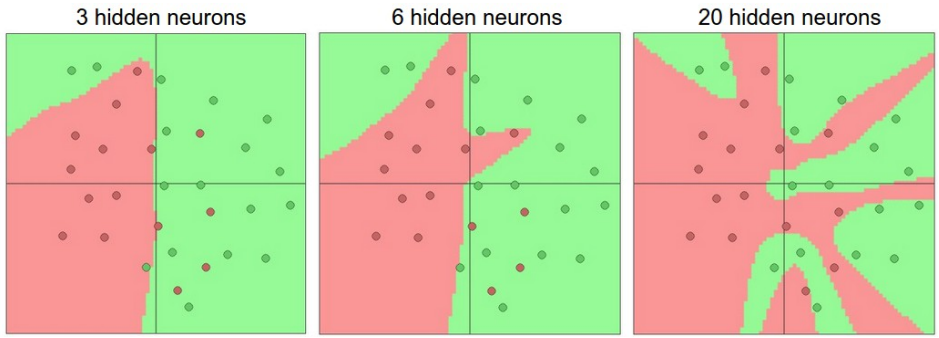
- Cuanto más complejo es un sistema de aprendizaje, más probable es que *sobreaprenda* (memorize cada uno de los ejemplos de entrenamiento de forma individual), y que sea incapaz de generalizar.
- Un sistema demasiado simple no puede representar distribuciones complejas de los datos, pero generaliza mucho más.

Complejidad y generalización

- Cuanto más complejo es un sistema de aprendizaje, más probable es que *sobreaprenda* (memorize cada uno de los ejemplos de entrenamiento de forma individual), y que sea incapaz de generalizar.
- Un sistema demasiado simple no puede representar distribuciones complejas de los datos, pero generaliza mucho más.
- Hay que encontrar un equilibrio entre la complejidad del modelo y su capacidad para reconocer los datos.

Complejidad y generalización

- Cuanto más complejo es un sistema de aprendizaje, más probable es que *sobreaprenda* (memorize cada uno de los ejemplos de entrenamiento de forma individual), y que sea incapaz de generalizar.
- Un sistema demasiado simple no puede representar distribuciones complejas de los datos, pero generaliza mucho más.
- Hay que encontrar un equilibrio entre la complejidad del modelo y su capacidad para reconocer los datos.



Entrenamiento de redes neuronales

Entrenamiento

- Una red viene dada por su topología (número de capas, neuronas, etc) y por los pesos asociados a cada neurona.

Entrenamiento de redes neuronales

Entrenamiento

- Una red viene dada por su topología (número de capas, neuronas, etc) y por los pesos asociados a cada neurona.
- La topología define la capacidad de aprendizaje y generalización.

Entrenamiento de redes neuronales

Entrenamiento

- Una red viene dada por su topología (número de capas, neuronas, etc) y por los pesos asociados a cada neurona.
- La topología define la capacidad de aprendizaje y generalización.
- El aprendizaje se hace calculando los pesos a partir de los ejemplos del conjunto de entrenamiento.

Entrenamiento de redes neuronales

Entrenamiento

- Una red viene dada por su topología (número de capas, neuronas, etc) y por los pesos asociados a cada neurona.
- La topología define la capacidad de aprendizaje y generalización.
- El aprendizaje se hace calculando los pesos a partir de los ejemplos del conjunto de entrenamiento.

El algoritmo *backpropagation*

- Se calcula la salida de la red para una muestra en concreto.

Entrenamiento de redes neuronales

Entrenamiento

- Una red viene dada por su topología (número de capas, neuronas, etc) y por los pesos asociados a cada neurona.
- La topología define la capacidad de aprendizaje y generalización.
- El aprendizaje se hace calculando los pesos a partir de los ejemplos del conjunto de entrenamiento.

El algoritmo *backpropagation*

- Se calcula la salida de la red para una muestra en concreto.
- Se calcula la diferencia entre la salida esperada y la salida obtenida.

Entrenamiento de redes neuronales

Entrenamiento

- Una red viene dada por su topología (número de capas, neuronas, etc) y por los pesos asociados a cada neurona.
- La topología define la capacidad de aprendizaje y generalización.
- El aprendizaje se hace calculando los pesos a partir de los ejemplos del conjunto de entrenamiento.

El algoritmo *backpropagation*

- Se calcula la salida de la red para una muestra en concreto.
- Se calcula la diferencia entre la salida esperada y la salida obtenida.
- Se actualizan los pesos *hacia atrás*.

Entrenamiento de redes neuronales

Entrenamiento

- Una red viene dada por su topología (número de capas, neuronas, etc) y por los pesos asociados a cada neurona.
- La topología define la capacidad de aprendizaje y generalización.
- El aprendizaje se hace calculando los pesos a partir de los ejemplos del conjunto de entrenamiento.

El algoritmo *backpropagation*

- Se calcula la salida de la red para una muestra en concreto.
- Se calcula la diferencia entre la salida esperada y la salida obtenida.
- Se actualizan los pesos *hacia atrás*.
- La actualización se hace de forma progresiva, en un proceso de optimización, en el que se revisan todas las muestras, agrupadas en pequeños lotes *(batchs)*, un número determinado de veces *(epochs)*.

Redes neuronales convolucionales

- Toman como entrada datos N-dimensionales (N suele ser 2 o 3).

Redes neuronales convolucionales

- Toman como entrada datos N-dimensionales (N suele ser 2 o 3).
- Poseen neuronas que se comportan como máscaras de convolución.

Redes neuronales convolucionales

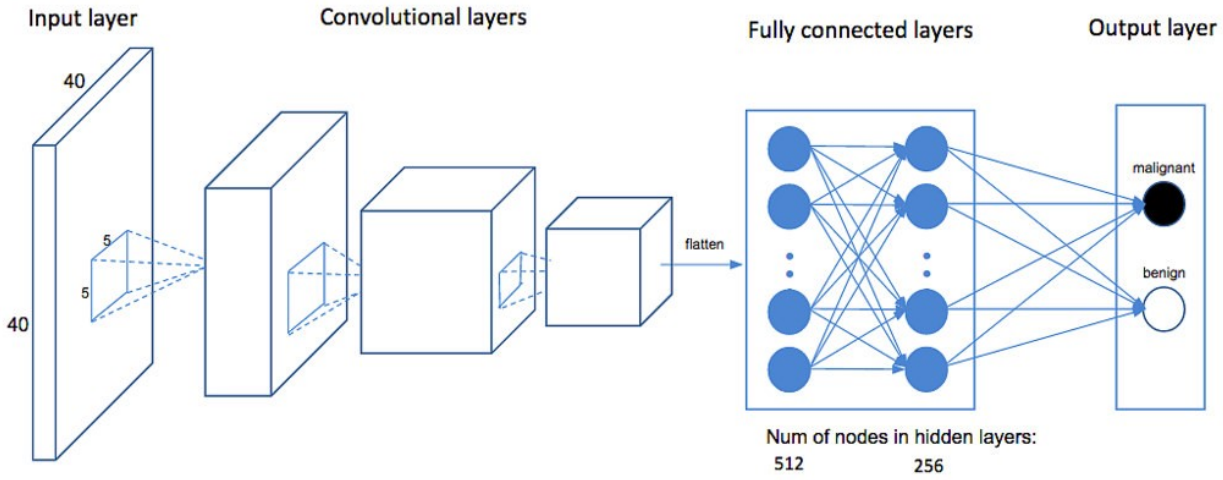
- Toman como entrada datos N-dimensionales (N suele ser 2 o 3).
- Poseen neuronas que se comportan como máscaras de convolución.
- Poseen capas que reducen el tamaño de los datos.

Redes neuronales convolucionales

- Toman como entrada datos N-dimensionales (N suele ser 2 o 3).
- Poseen neuronas que se comportan como máscaras de convolución.
- Poseen capas que reducen el tamaño de los datos.
- La última o últimas capas son totalmente conectadas.

Redes neuronales convolucionales

- Toman como entrada datos N-dimensionales (N suele ser 2 o 3).
- Poseen neuronas que se comportan como máscaras de convolución.
- Poseen capas que reducen el tamaño de los datos.
- La última o últimas capas son totalmente conectadas.



http://cs.stanford.edu/people/karpathy/convnetjs/

Capas de convolución

- Una neurona de convolución está asociada a una posición de la imagen y está conectada con los píxeles de su vecindario, de forma que sus pesos son los coeficientes de la máscara de convolución.

Capas de convolución

- Una neurona de convolución está asociada a una posición de la imagen y está conectada con los píxeles de su vecindario, de forma que sus pesos son los coeficientes de la máscara de convolución.
- Cada juego de coeficientes se denomina *filtro*. Se usa el mismo para todas las posiciones de la imagen.

Capas de convolución

- Una capa de convolución que tome como entrada una imagen en color de $M \times N \times 3$ valores, con F filtros de tamaño $H \times H \times 3$, tiene el siguiente número de coeficientes, contando el sesgo:

$$F \times (H \times H \times 3 + 1)$$

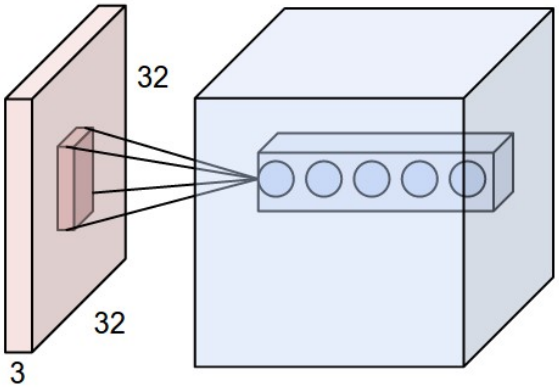
y genera como salida un volumen de $M' \times N' \times F$ coeficientes, siendo M' y N' el ancho y el alto de la imagen menos los bordes.

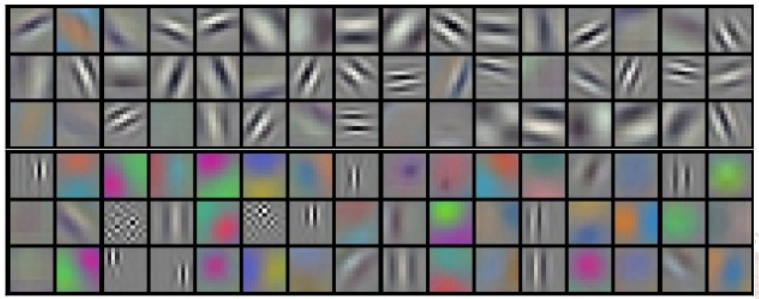
Capas de convolución

- Una capa de convolución que tome como entrada una imagen en color de $M \times N \times 3$ valores, con F filtros de tamaño $H \times H \times 3$, tiene el siguiente número de coeficientes, contando el sesgo:

$$F \times (H \times H \times 3 + 1)$$

y genera como salida un volumen de $M' \times N' \times F$ coeficientes, siendo M' y N' el ancho y el alto de la imagen menos los bordes.





Capas de *max pooling*

- Reducen el tamaño de la imagen, tomando máximos por áreas.

Capas de *max pooling*

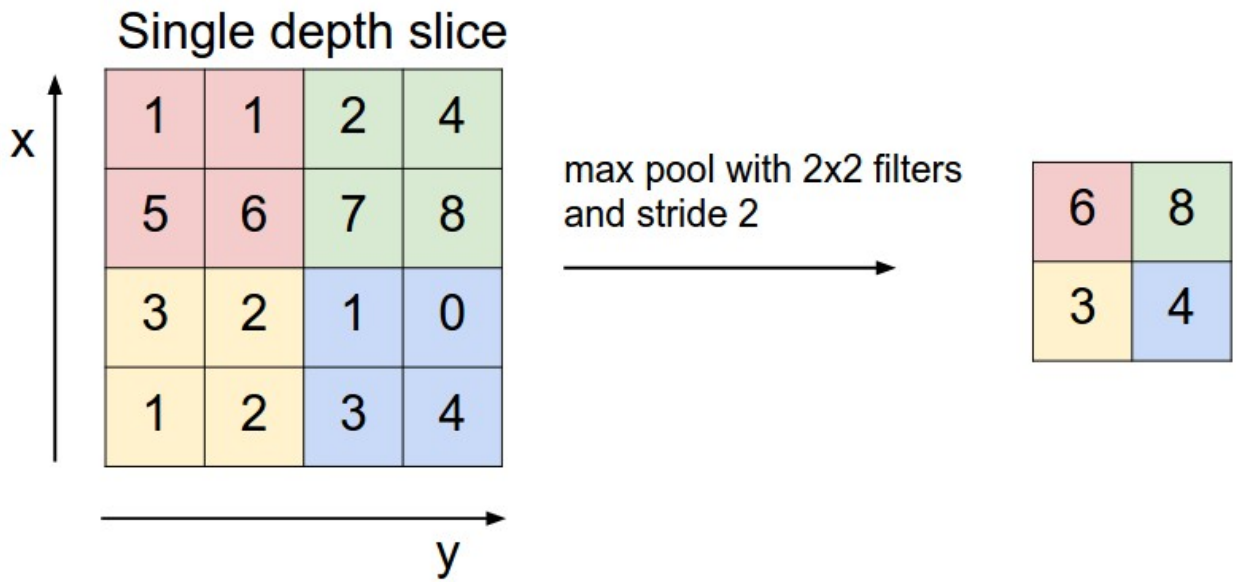
- Reducen el tamaño de la imagen, tomando máximos por áreas.
- No tienen coeficientes que entrenar.

Capas de *max pooling*

- Reducen el tamaño de la imagen, tomando máximos por áreas.
- No tienen coeficientes que entrenar.
- Al aplicar *backpropagation*, solo se propagan los valores hacia el lugar donde estuviera el máximo cuando se evaluó la red para el ejemplo.

Capas de *max pooling*

- Reducen el tamaño de la imagen, tomando máximos por áreas.
- No tienen coeficientes que entrenar.
- Al aplicar *backpropagation*, solo se propagan los valores hacia el lugar donde estuviera el máximo cuando se evaluó la red para el ejemplo.



Capas de *dropout*

- Anulan aleatoriamente una serie de neuronas al entrenar cada ejemplo.

Capas de *dropout*

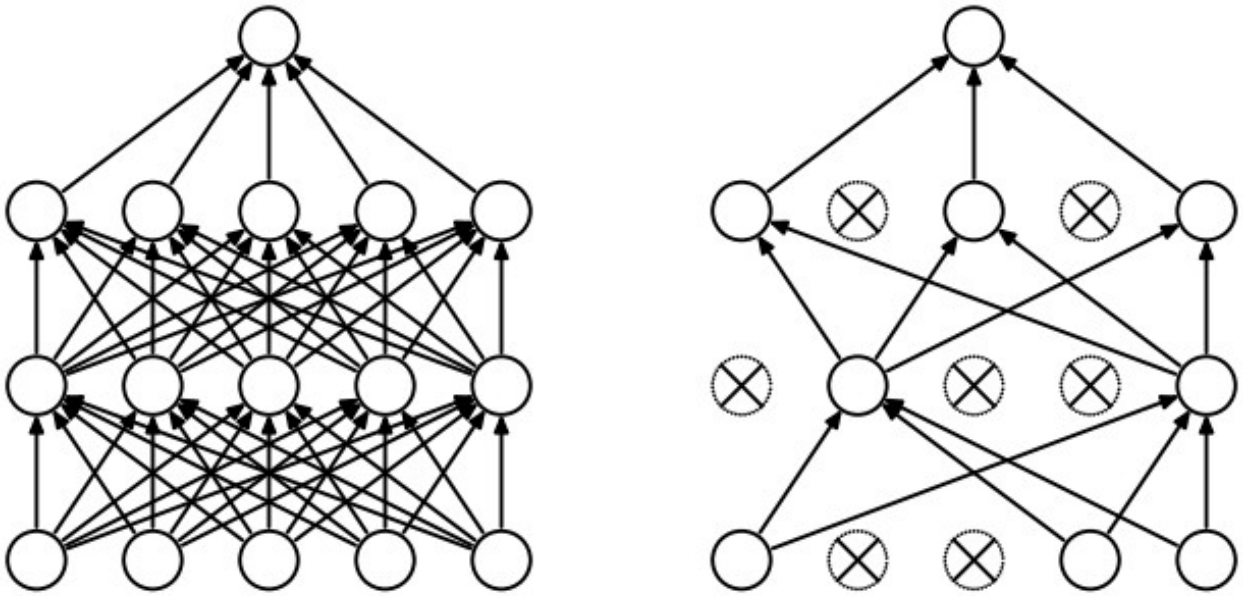
- Anulan aleatoriamente una serie de neuronas al entrenar cada ejemplo.
- Hacen que las neuronas aprendan de forma independiente, y reducen el tiempo de entrenamiento.

Capas de *dropout*

- Anulan aleatoriamente una serie de neuronas al entrenar cada ejemplo.
- Hacen que las neuronas aprendan de forma independiente, y reducen el tiempo de entrenamiento.
- No tienen coeficientes que entrenar.

Capas de *dropout*

- Anulan aleatoriamente una serie de neuronas al entrenar cada ejemplo.
- Hacen que las neuronas aprendan de forma independiente, y reducen el tiempo de entrenamiento.
- No tienen coeficientes que entrenar.



(a) Standard Neural Net (b) After applying dropout.

Capas de *batch normalization*

- Permiten normalizar (y en parte regularizar) los datos.

Capas de *batch normalization*

- Permiten normalizar (y en parte regularizar) los datos.
- A todos los datos de un lote x_i se les resta la media y se dividen por su desviación típica, y se obtiene $\hat{x}_i$.

Capas de *batch normalization*

- Permiten normalizar (y en parte regularizar) los datos.
- A todos los datos de un lote x_i se les resta la media y se dividen por su desviación típica, y se obtiene $\hat{x}_i$.
- La salida de la capa es $y_i = \lambda \hat{x}_i + \beta$.

Capas de *batch normalization*

- Permiten normalizar (y en parte regularizar) los datos.
- A todos los datos de un lote x_i se les resta la media y se dividen por su desviación típica, y se obtiene $\hat{x}_i$.
- La salida de la capa es $y_i = \lambda \hat{x}_i + \beta$.
- Los parámetros λ y β son aprendidos por la red.

Algunos tipos de redes profundas

- Redes recurrentes.

Algunos tipos de redes profundas

- Redes recurrentes.
- *Wavenets*.

Algunos tipos de redes profundas

- Redes recurrentes.
- *Wavenets*.
- Codificadores *(encoders)* y decodificadores *(decoders)*.

Algunos tipos de redes profundas

- Redes recurrentes.
- *Wavenets*.
- Codificadores *(encoders)* y decodificadores *(decoders)*.
- Redes generativas y adversarias *(GANS)*.

Algunos tipos de redes profundas

- Redes recurrentes.
- *Wavenets*.
- Codificadores *(encoders)* y decodificadores *(decoders)*.
- Redes generativas y adversarias *(GANS)*.
- *Long short-term memory (LTSM)*.

Identificación de caras mediante *deep learning*

Deep Face

- Emplea redes neuronales *profundas*:

Identificación de caras mediante *deep learning*

Deep Face

- Emplea redes neuronales *profundas*:
  - 9 capas, con 120 millones de coeficientes.

Identificación de caras mediante *deep learning*

Deep Face

- Emplea redes neuronales *profundas*:
  - 9 capas, con 120 millones de coeficientes.
- Alineación de la cara (preprocesamiento):

Identificación de caras mediante *deep learning*

Deep Face

- Emplea redes neuronales *profundas*:
  - 9 capas, con 120 millones de coeficientes.
- Alineación de la cara (preprocesamiento):
  - Detección mediante clasificadores débiles.

Identificación de caras mediante *deep learning*

Deep Face

- Emplea redes neuronales *profundas*:
  - 9 capas, con 120 millones de coeficientes.
- Alineación de la cara (preprocesamiento):
  - Detección mediante clasificadores débiles.
  - Localización de puntos *singulares*.

Identificación de caras mediante *deep learning*

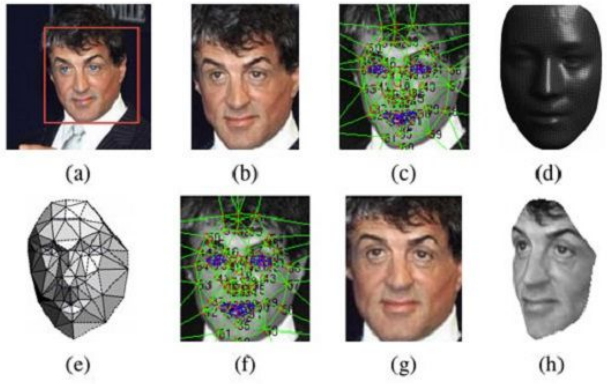
Deep Face

- Emplea redes neuronales *profundas*:
  - 9 capas, con 120 millones de coeficientes.
- Alineación de la cara (preprocesamiento):
  - Detección mediante clasificadores débiles.
  - Localización de puntos *singulares*.
  - *Estirado* de la imagen para eliminar el efecto de la perspectiva *(frontalización)*.

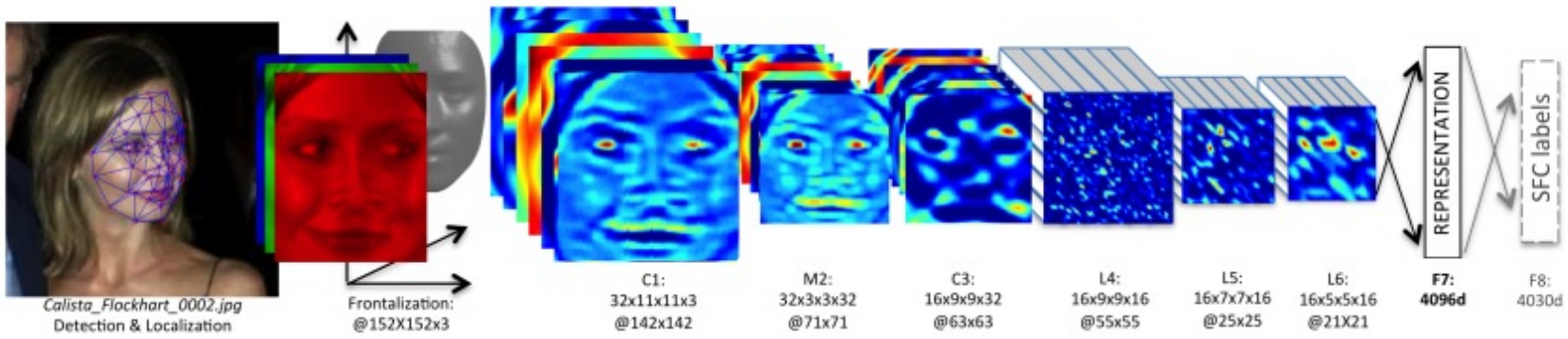
Identificación de caras mediante *deep learning*

Deep Face

- Emplea redes neuronales *profundas*:
  - 9 capas, con 120 millones de coeficientes.
- Alineación de la cara (preprocesamiento):
  - Detección mediante clasificadores débiles.
  - Localización de puntos *singulares*.
  - *Estirado* de la imagen para eliminar el efecto de la perspectiva *(frontalización)*.

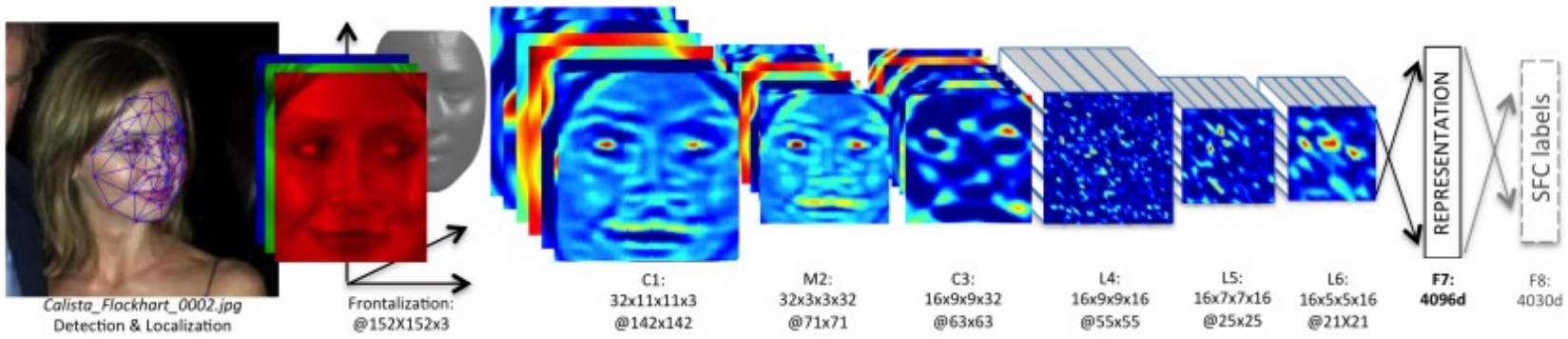


Identificación de caras mediante *deep learning*



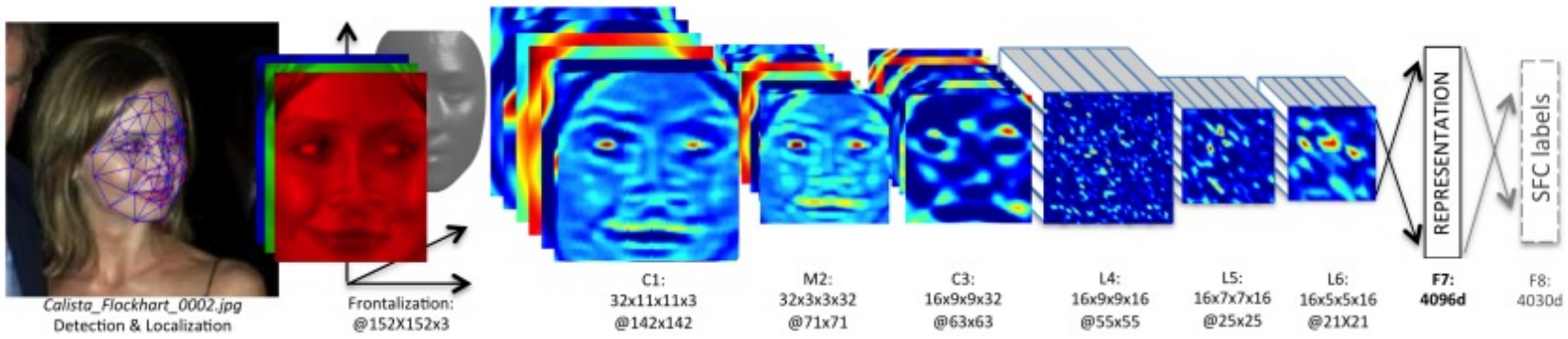
- Estructura:

Identificación de caras mediante *deep learning*



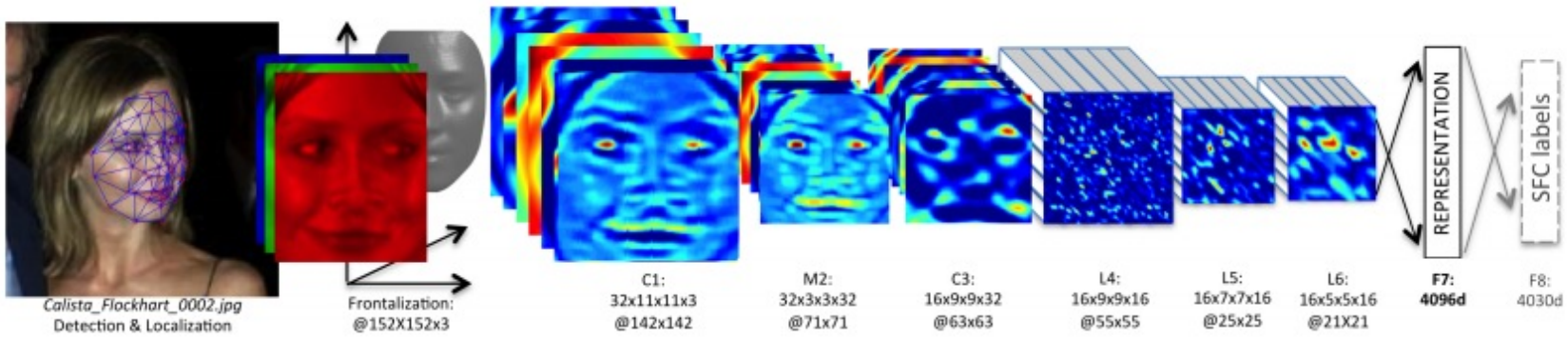
- Estructura:
  - Dos capas *convolucionales* (C1 y C3).

Identificación de caras mediante *deep learning*



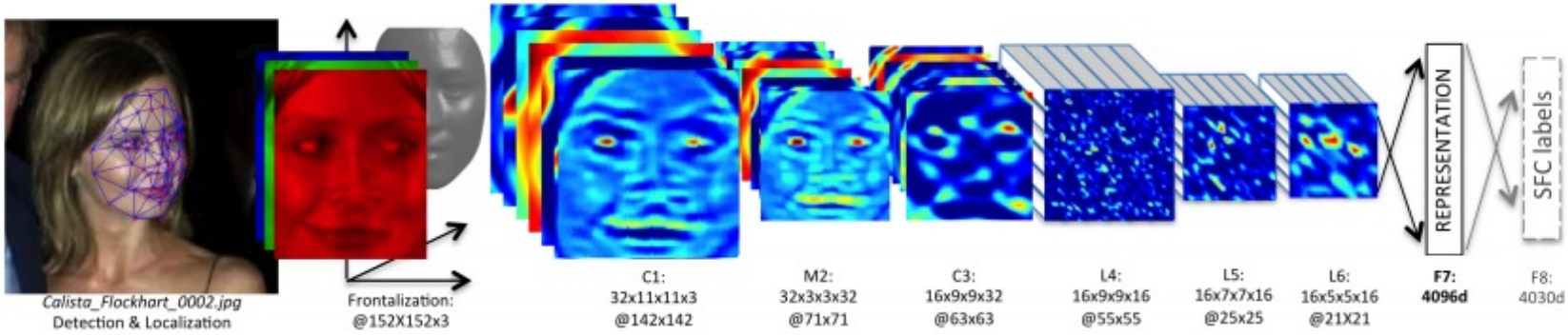
- Estructura:
  - Dos capas *convolucionales* (C1 y C3).
  - Una capa de *max pooling* (M2).

Identificación de caras mediante *deep learning*



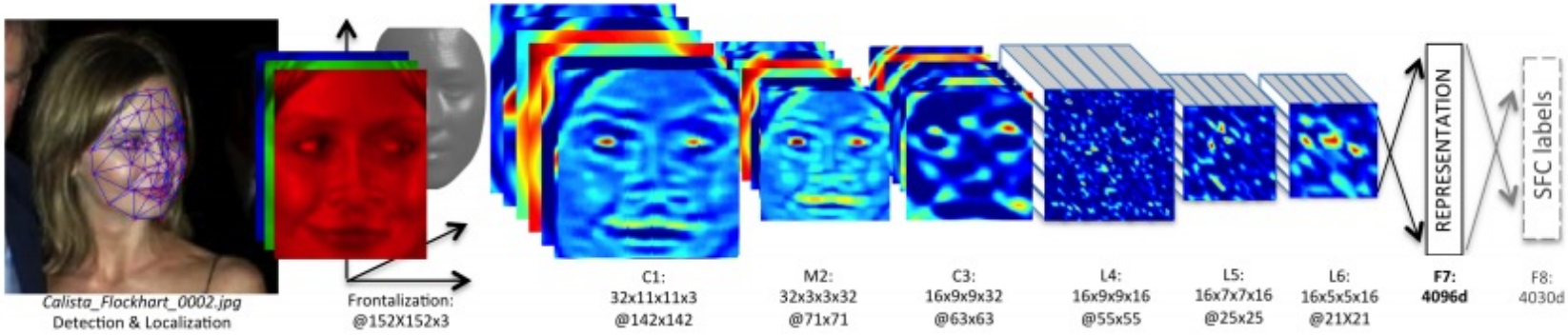
- Estructura:
  - Dos capas *convolucionales* (C1 y C3).
  - Una capa de *max pooling* (M2).
  - Tres capas localmente conectadas (L4, L5 y L6).

Identificación de caras mediante *deep learning*



- Estructura:
  - Dos capas *convolucionales* (C1 y C3).
  - Una capa de *max pooling* (M2).
  - Tres capas localmente conectadas (L4, L5 y L6).
  - Dos capas totalmente conectadas.

Identificación de caras mediante *deep learning*



- Estructura:
  - Dos capas *convolucionales* (C1 y C3).
  - Una capa de *max pooling* (M2).
  - Tres capas localmente conectadas (L4, L5 y L6).
  - Dos capas totalmente conectadas.
- Entrenada con 4 millones de ejemplos, obtiene una tasa de acierto del 97.35 %.

Recursos sobre redes profundas

- https://github.com/celiacintas/star_wars_hackathon
- https://cs.stanford.edu/people/karpathy/convnetjs/
- http://playground.tensorflow.org/
- http://www.deeplearningbook.org/