

## Percepción automática

- Extracción de características y segmentación de imágenes
- Reconocimiento de objetos

## Índice

- Transformada de Hough
- Características SIFT
- Segmentación de imágenes
- Algoritmo de las K-medias
- Segmentación basada en regiones
- Reconocimiento de objetos
- Reconocimiento mediante características
- Reconocimiento de caras

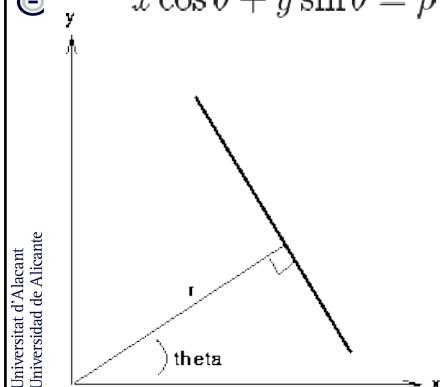
## Transformada de Hough

- Fundamentos:
  - Motivación:
    - Desarrollar técnicas que permitan identificar primitivas geométricas sencillas (líneas, círculos, cuadrados....) en la imagen (p.e. tras binarizar el resultado de Canny).
  - Mecanismo:
    - Espacio paramétrico. Los valores de los parámetros de la ecuación paramétrica definen únicamente a cada primitiva. Por cada parámetro tenemos una dimensión en el espacio paramétrico y cada dimensión se “discretiza” en celdas.
    - Proceso de “votación”. Cada punto de la imagen votará por aquella ecuación (combinación de parámetros, o celda) que cumpla. Daremos como salida las celdas con suficiente soporte o evidencia, esto es, aquellas con un “número suficiente de votos”.

## Transformada de Hough

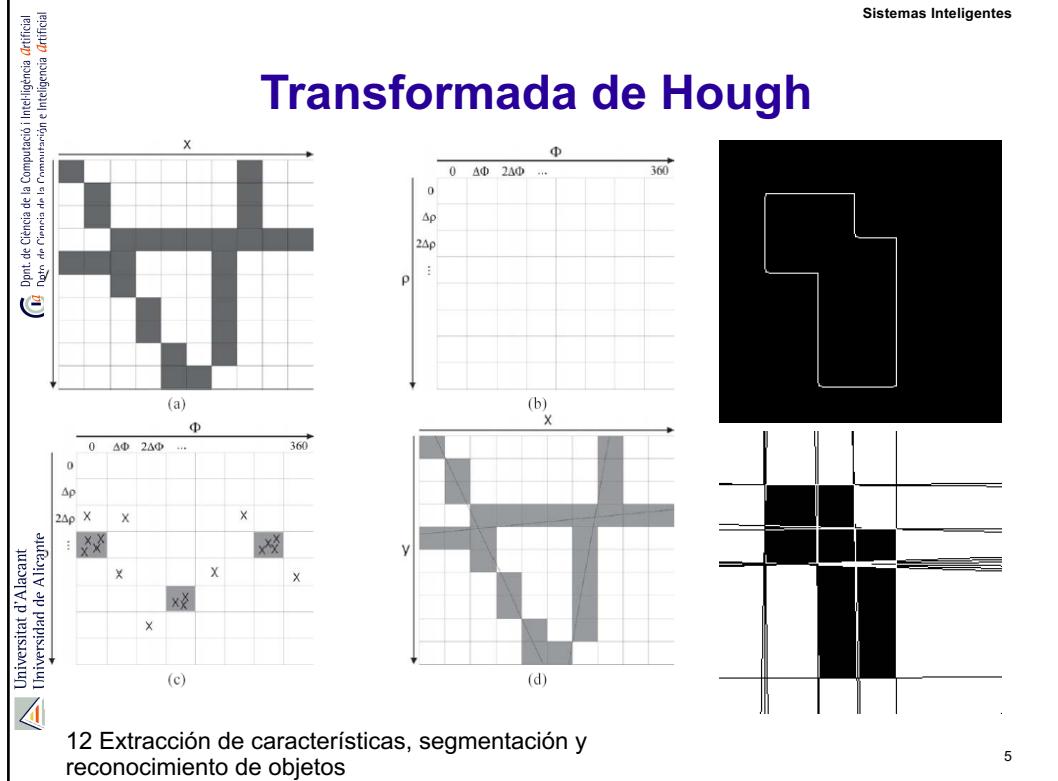
- Transformada para la recta:
  - Ecuación paramétrica:

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



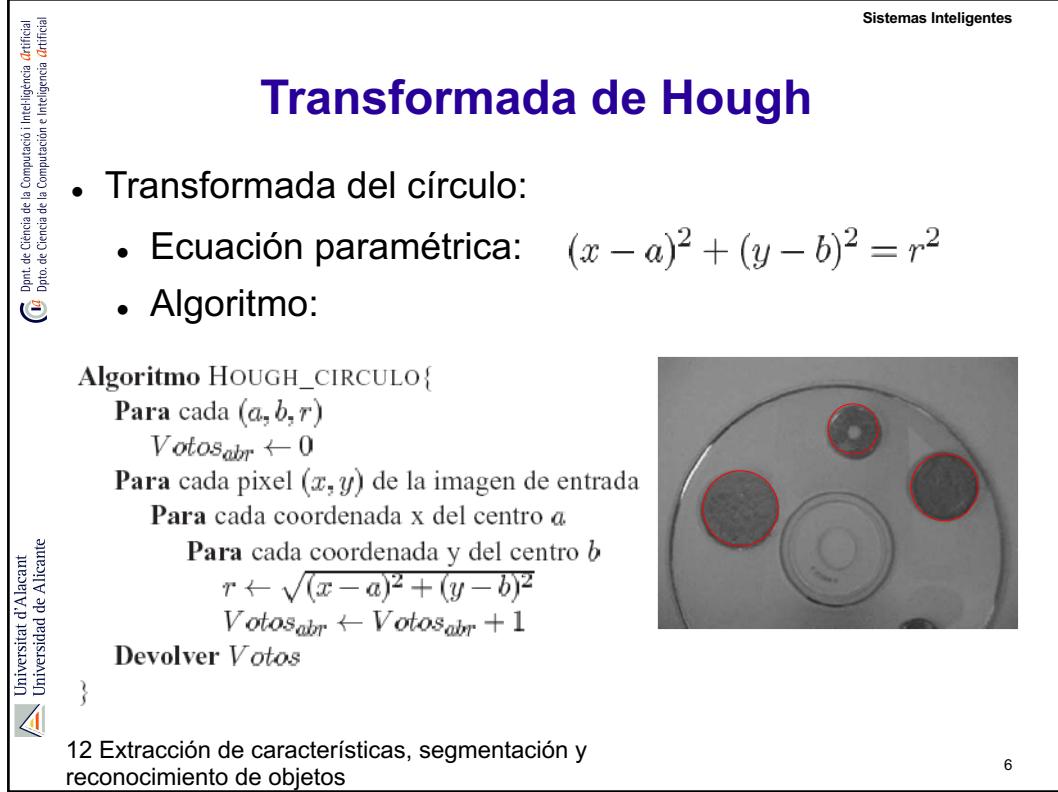
```
Algoritmo HOUGH_RECTA {
  Para cada  $(\rho, \theta)$ 
     $Votos_{\rho\theta} \leftarrow 0$ 
  Para cada pixel  $(x, y)$  de la imagen de entrada
    Para cada orientación posible de las rectas  $\theta$ 
       $\rho \leftarrow x \cos \theta + y \sin \theta$ 
       $Votos_{\rho\theta} \leftarrow Votos_{\rho\theta} + 1$ 
    Devolver  $Votos$ 
}
```

## Transformada de Hough



5

## Transformada de Hough



6

## Transformada de Hough

- Análisis:

- Puntos críticos:
  - Ineficiencia: la complejidad espacial de la construcción del espacio de parámetros aumenta de forma factorial con el número de los mismos:  $O(N!)$  Lo mismo sucede con la complejidad temporal del proceso de votación:  $O(N-1)!$
  - En consecuencia, el método exhaustivo solamente es aplicable a primitivas sencillas (caracterizadas por pocos parámetros) e incluso en estos casos se hace necesaria una discretización muy somera para satisfacer los requerimientos de memoria.
  - Ajuste de sensibilidad: aunque es un factor de menor importancia, el ajuste del tamaño de celda y del número mínimo de votos (a menudo seguido de una supresión de no-máximos) es clave para evitar un elevado número de falsos positivos.

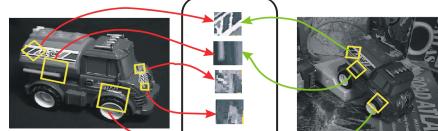
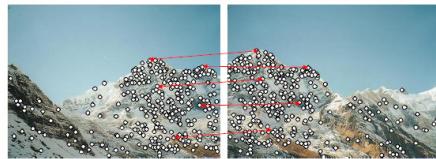
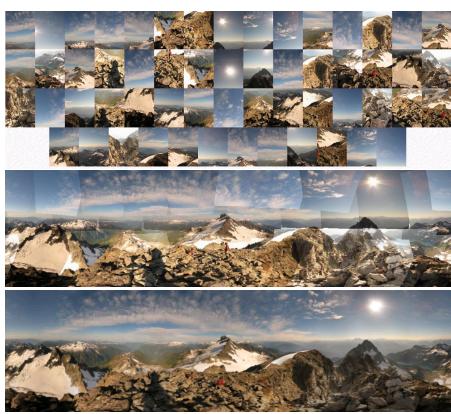


## Características “extendidas”

- Las características de una imagen se puede entender como una representación “comprendida” de la imagen, pero reducida
- Existen una serie de tareas en visión que necesitan extraer estas características
  - Reconocimiento de objetos
  - Reconstrucción 3D
  - Seguimiento de movimiento
  - Búsqueda en base de datos de imágenes
  - Etc.
- Estas características necesitarán ser identificadas de manera “única”



## Ejemplos



- <http://cvlab.epfl.ch/~brown/autostitch/autostitch.html>
- Problemas: cambios de iluminación, tamaño, punto de vista, occlusiones, etc.

12 Extracción de características, segmentación y reconocimiento de objetos

9

## Solución

- Usar, además de la posición en la imagen, algún “descriptor” que identifique el punto de manera “única” (o casi única)
- El descriptor contendrá algún tipo de información de la imagen alrededor del punto
- La misma característica, vista desde distintos puntos de vista, con distinta iluminación, con giros, etc. tendrá el mismo descriptor.
- Su cálculo debe ser eficiente.
- Existen varias alternativas: SIFT, SURF, MSER, etc.

12 Extracción de características, segmentación y reconocimiento de objetos

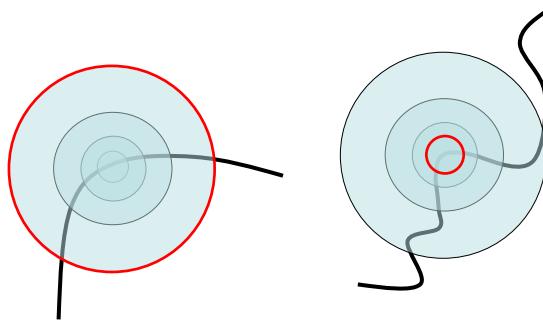
10

# SIFT

- David G. Lowe, "Distinctive image features from scale-invariant keypoints," International Journal of Computer Vision, 60, 2 (2004)
- SIFT: Scale Invariant Feature Transform
- Uno de los primeros en ser usados: el autor deja un ejecutable que puede ser usado
- El algoritmo devuelve las características encontradas en la imagen
- Tiene dos pasos:
  - Encontrar la posición de los puntos (se podría usar Harris o cualquier otro)
  - Calcular el descriptor

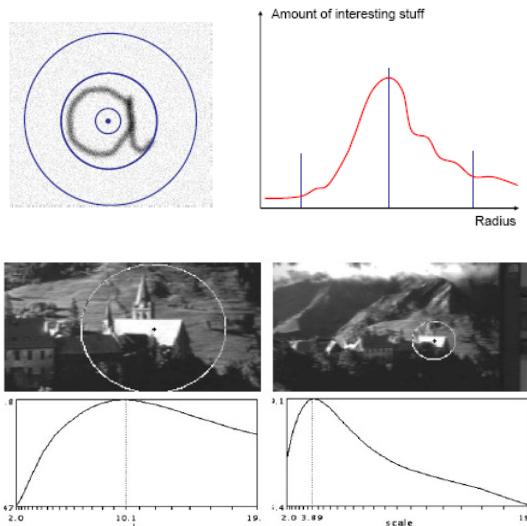
# Localización de las características

- Harris podría ser un buen candidato
  - ¿Por qué?
  - ¿Y en cuanto a la escala?
- La idea es intentar que el detector sea invariante a escala. Sift propone una detección multiescala



## Localización multiescala

- ¿Cómo se puede elegir la escala?

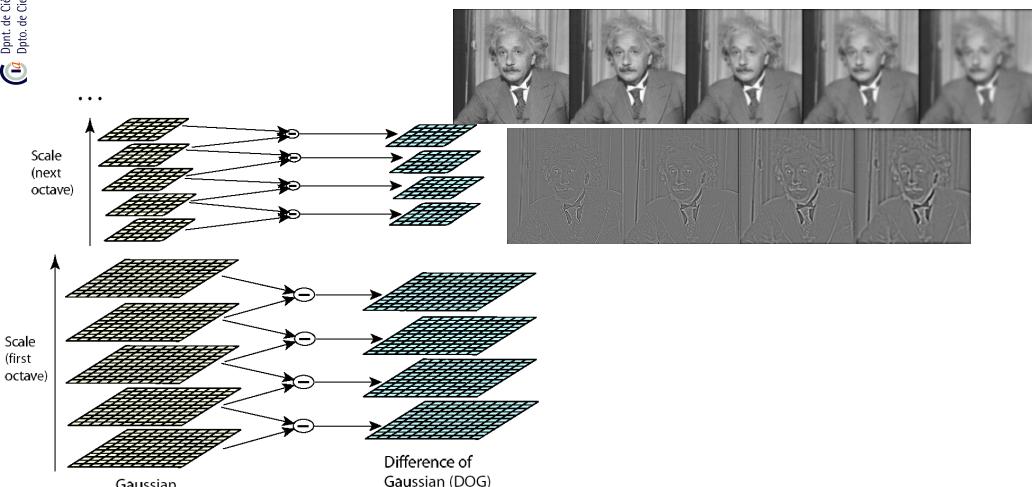


12 Extracción de características, segmentación y reconocimiento de objetos

13

## Localización multiescala

- La localización en la multiescala se consigue con una diferencia de gaussianas (DoG)

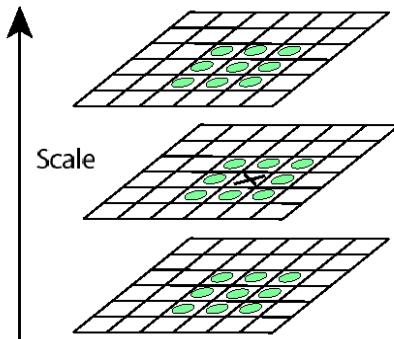


12 Extracción de características, segmentación y reconocimiento de objetos

14

## Localización

- Hay que encontrar un mínimo o máximo entre escalas



## Eliminación de puntos inestables

- Se encuentran muchos puntos, pero muchos de ellos son “inestables”
- Se eliminan aquellos:
  - Con un contraste bajo
  - Que se encuentran sobre una arista
- Iniciales: 832. Finales: 536

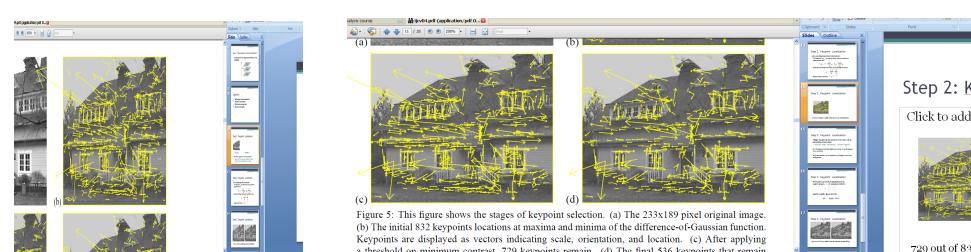
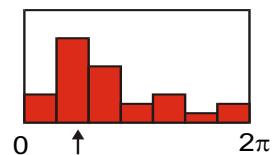
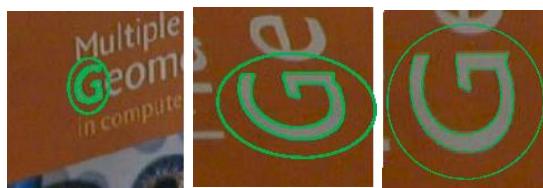
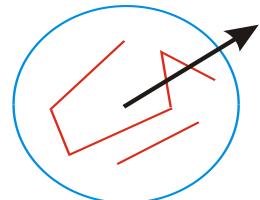


Figure 5. This figure shows the stages of keypoint selection. (a) The 233x159 pixel original image. (b) The initial 832 keypoints locations at maxima and minima of the difference-of-Gaussian function. Keypoints are displayed as vectors indicating scale, orientation, and location. (c) After applying a threshold on minimum contrast, 729 keypoints remain. (d) The final 536 keypoints that remain

## Orientación de la característica

- Es necesario encontrar cómo está orientada la característica
- Se encuentra el histograma de orientaciones **ponderado en la escala calculada** y se calcula la orientación dominante

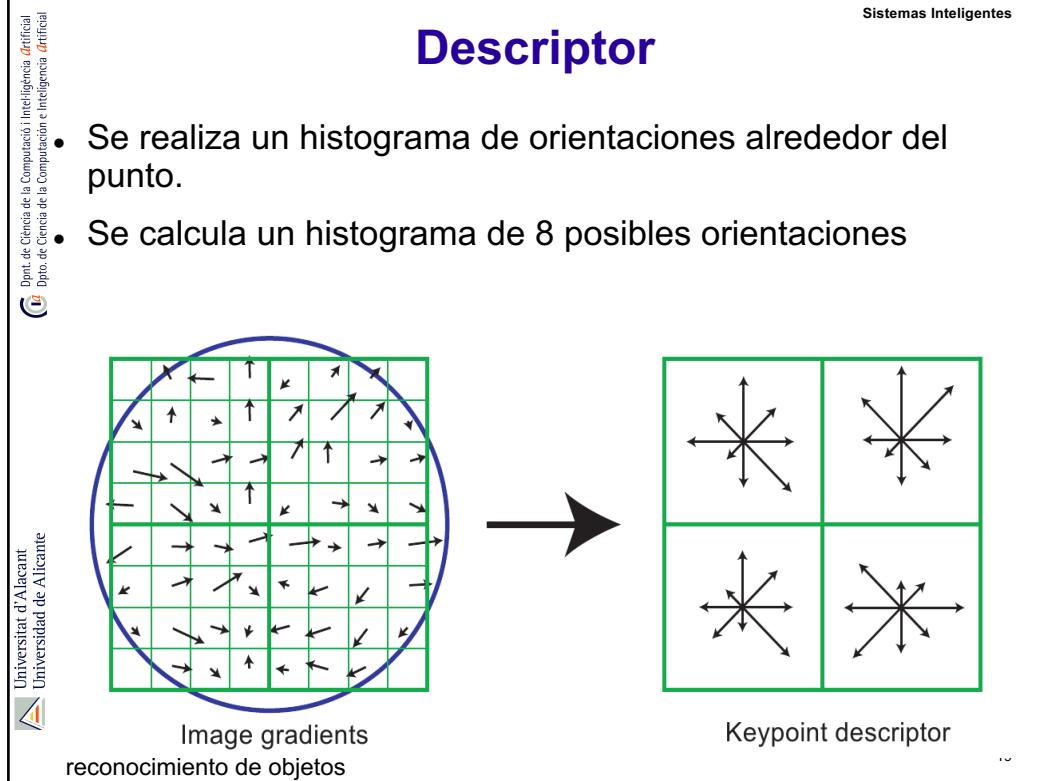


## Descriptor

- Hasta el momento hemos calculado:
  - Localización del punto.
  - Escala.
  - Orientación.
- Debemos encontrar el descriptor que nos permita comparar entre las características
- Se escoge una cierta vecindad del punto a la escala determinada
- Debemos hacer una “rectificación” de la vecindad para luego calcular descriptor

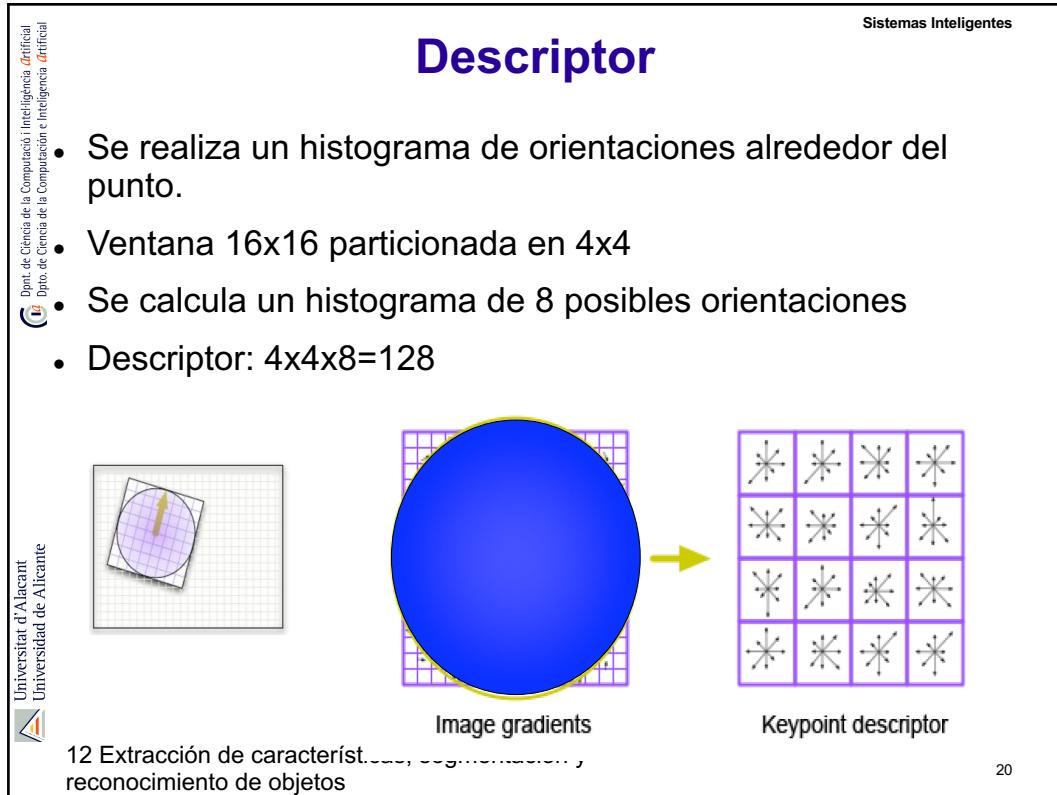
## Descriptor

- Se realiza un histograma de orientaciones alrededor del punto.
- Se calcula un histograma de 8 posibles orientaciones

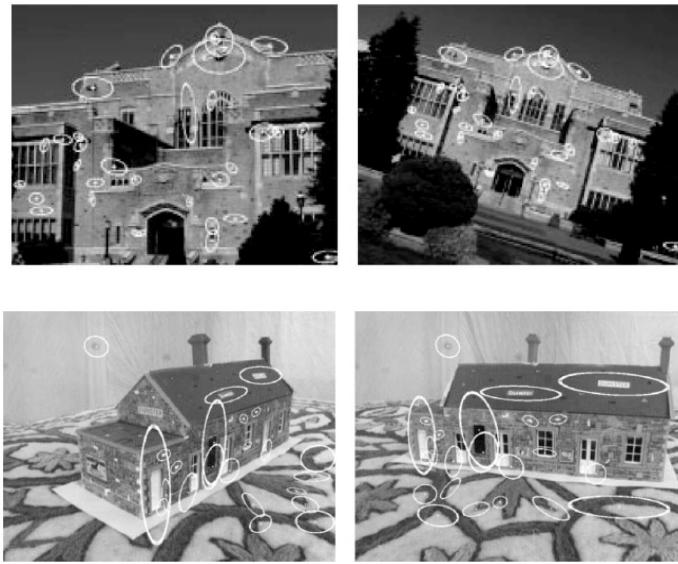


## Descriptor

- Se realiza un histograma de orientaciones alrededor del punto.
- Ventana 16x16 particionada en 4x4
- Se calcula un histograma de 8 posibles orientaciones
- Descriptor:  $4 \times 4 \times 8 = 128$



## Ejemplos

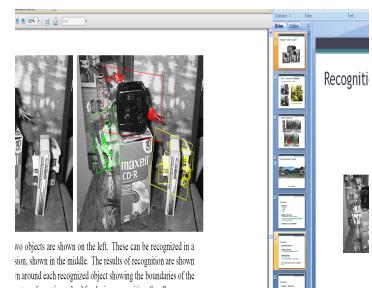
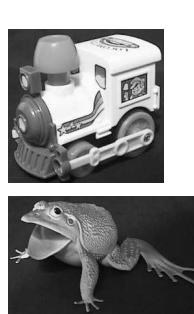


12 Extracción de características, segmentación y reconocimiento de objetos

21



## Reconocimiento en oclusión

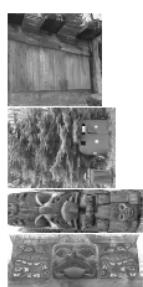


No objects are shown on the left. These can be recognized in a sit, shown in the middle. The results of recognition are shown in around each recognized object showing the boundaries of the transformation solved for doing recognition. Smaller sources

12 Extracción de características, segmentación y reconocimiento de objetos

22

## Otro ejemplo

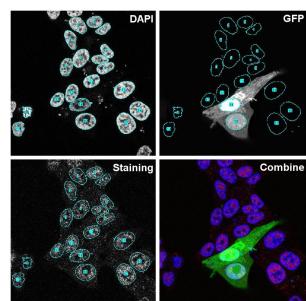
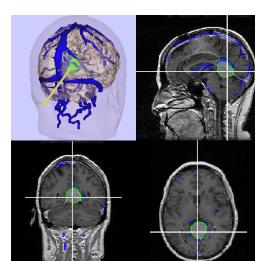
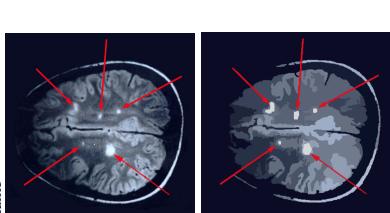


12 Extracción de características, segmentación y reconocimiento de objetos

23

## Segmentación de imágenes

- La segmentación de imágenes es el proceso de extraer zonas de la imagen con el mismo color/nivel de gris/textura para identificarlas automáticamente

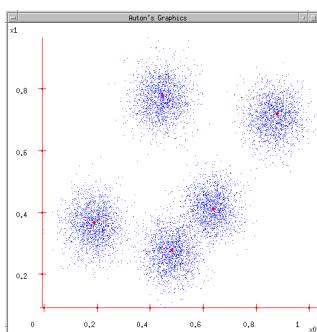


12 Extracción de características, segmentación y reconocimiento de objetos

24

## Algoritmo de las K-medias

- Encuentra las medias de las distribuciones (*clusters*) (sirve para cualquier problema de aprendizaje)
- Es necesario conocer el número de distribuciones (*clusters*) distintas existentes



12 Extracción de características, segmentación y reconocimiento de objetos

25

## Algoritmo de las K-medias

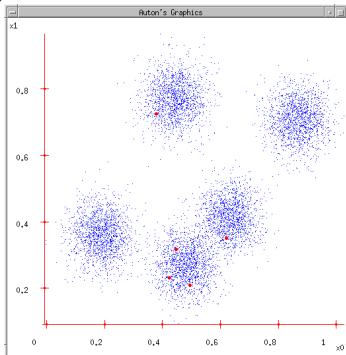
- 1) Inicialización: se buscan  $k$  puntos (medias)
- 2) Hacer
  - 1) Pertenencia: para cada punto, se encuentra la distribución (*cluster*) más cercana (el  $k$  más cercano)
  - 2) Cálculo de las nuevas medias
- 3) Hasta que no haya cambios

12 Extracción de características, segmentación y reconocimiento de objetos

26

## Inicialización

- Para elegir la inicialización, podemos:
  - Asignar las medias de manera aleatoria.
  - Redistribuir las medias de manera uniforme
  - Usar alguna heurística (por ejemplo, en secuencias de imágenes, usar las medias resultado anteriores)



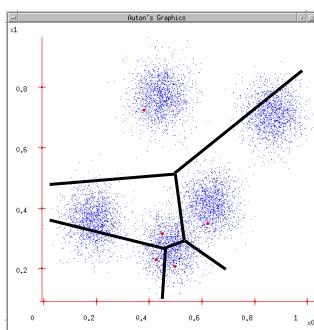
12 Extracción de características, segmentación y reconocimiento de objetos

27

## Cálculo de la pertenencia

- Dependerá del problema:
  - Distancia euclídea: el  $k$  más cercano
  - Probabilidad de pertenencia a un cluster: usando su media y varianza

$$P(x) = \frac{1}{\sqrt{\pi\sigma^2}} \exp\left(-\frac{(x-\mu)^2}{\sigma^2}\right)$$



12 Extracción de características, segmentación y reconocimiento de objetos

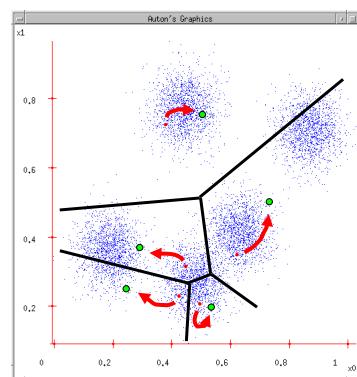
28

## Cálculo de las nuevas medias

- Para cada cluster, recalcular su media (y varianza, si es el caso), usando todos los puntos que pertenecen a ese cluster

- Si es distancia euclídea:  $\mu_{cluster} = \frac{1}{n} \sum_{i=1}^n x_i$

- Si es probabilidad:  $\mu_{cluster} = \frac{\sum_{i=1}^n p(x_i) x_i}{\sum_{i=1}^n p(x_i)}$



12 Extracción de características, segmentación y reconocimiento de objetos

29

## K-medias para segmentación de imágenes

- Imágenes de gris: se usan los valores de gris, no las posiciones de los píxeles.
- Imágenes de color:
  - Usar directamente los valores R,G,B
  - Convertir a H,S,B y usar la componente H



(probabilidad K=4 112ms)

## Características de las K-medias

- Es necesario indicar K
- Una mala inicialización puede llevar más tiempo
- Puede no encontrar la solución más óptima
- En su modo probabilístico:
  - Se puede aplicar a cualquier tipo de distribución (texturas)
  - Se pueden usar distintas medidas de distancia a distribución (Mahalanobis, Kullback-Leiber)
  - Se conoce también como: segmentación EM, Fuzzy K-means



## Segmentación basada en regiones

- El objetivo es encontrar regiones de la imagen homogéneas según algún criterio
- Hay dos maneras:
  - Crecimiento de regiones (*region growing*): empezamos con regiones pequeñas (semillas) y las hacemos crecer o bien las mezclamos, usando un criterio de similaridad
  - Partición de regiones (*region splitting*): empezamos con regiones grandes (incluso toda la imagen) y las vamos dividiendo usando un criterio de homogeneidad
- Existen métodos que combinan las dos maneras



## Crecimiento de regiones

- Haralick and Sapiro "Image segmentation techniques" Computer Vision, Graphics and Image Processing. V. 29 (1). 1985
- Asume que una región es un conjunto de píxeles conectados
- Sea  $R$  una región de  $N$  píxeles

$$\mu = \frac{1}{N} \sum_{x_i \in R} I(x_i)$$

$$\sigma^2 = \sum_{x_i \in R} (I(x_i) - \mu)^2$$

- Tenemos que definir un test de similaridad para saber si un nuevo píxel  $y$ , adyacente a algún píxel de  $R$ , se puede añadir a la región



## Crecimiento de regiones: test de similaridad

- Podemos definir el siguiente test:

- Calculamos  $T$ :

$$T = \sqrt{\frac{(N-1)N}{(N+1)} \frac{(y - \mu)^2}{\sigma^2}}$$

N: número de elementos que forman la región  
 $\mu$ : valor medio de la región  
 $\sigma^2$ : varianza de la región

- Si  $T$  es lo suficientemente pequeño, añadimos  $y$  a  $R$
  - Si se añade, debemos actualizar la media y varianza:

$$\mu_{nueva} = \frac{N\mu + y}{(N+1)} \quad \sigma_{nueva}^2 = \sigma^2 + (y - \mu_{nueva})^2 + N(\mu_{nueva} - \mu)^2$$

- Si no se añade, empezamos a calcular una **nueva** región con dicho píxel



## Crecimiento de regiones: semillas

- El método anterior lo podemos hacer empezando por cualquier píxel de la imagen
- Sin embargo, es mejor lanzar un conjunto de “semillas”, es decir, lanzar varios puntos de partida
- Para ello, se puede hacer lo siguiente:
  - Se aplica un detector de aristas
  - Los puntos cuyo valor de magnitud de gradiente estén próximos a cero, serán “valles”, es decir, puntos dentro de regiones
  - Usaremos esos puntos para hacer una “inundación” del valle (los usamos como “semillas” del método anterior)

## Partición de regiones

- Este proceso es el inverso del anterior
- Partimos una única región (toda la imagen) y usamos algún criterio de homogeneidad para partir la región
  - Si se cumple el criterio, no se sigue dividiendo
  - Si no se cumple, se divide
- Se puede usar el histograma para dicho criterio: ver si existen “valles”, que pueden servir a su vez para dividir la región en otras más pequeñas.
- Conlleva una complejidad mayor, al tener que manejar alguna estructura de datos adicional

# Percepción automática



## Reconocimiento de objetos

12 Extracción de características, segmentación y reconocimiento de objetos

37

# Índice



- Reconocimiento de objetos
- Reconocimiento mediante características
- Reconocimiento de caras

12 Extracción de características, segmentación y reconocimiento de objetos

38

## Introducción

- El reconocimiento de objetos consiste en, dado algún conocimiento (forma, apariencia, etc.) sobre uno o varios objetos y una imagen, encontrar qué objetos están en la imagen y dónde
- El reconocimiento es un proceso difícil debido a:
  - Presencia de otros objetos no modelados
  - Cambio de iluminación
  - Cambio de punto de vista del objeto
  - Oclusión
  - Escala



## Ejemplos: reconocimiento facial

- Ampliamente extendido: cámaras, Picasa
- Se suele reconocer una posible posición en la imagen para una cara (mediante color de la piel, identificación de los ojos, etc.) y luego se reconoce la persona (con técnicas de aprendizaje)

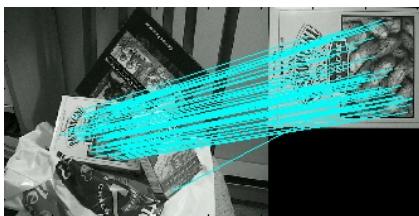


## Reconocimiento con características

- Imaginemos que tenemos una imagen de un objeto a reconocer (modelo)
- Extraemos, por ejemplo, las características SIFT de dicha imagen. El objeto ahora es representado por sus características SIFT
- Ahora tenemos una nueva imagen (escena) donde queremos “buscar” ese objeto
- Extraemos los SIFT de esta nueva imagen
- Encontramos las correspondencias entre las características del modelo y de la imagen

## Reconoc. Características: emparejamiento

- Esto se puede hacer calculando la distancia euclídea del descriptor
- Para cada característica del modelo:
  - Encontramos la característica de la escena cuya distancia euclídea esté por debajo de un cierto umbral
  - Ahora tenemos una correspondencia entre los descriptores del modelo y de la escena



## Reconocimiento características: transformación

- Ahora debemos encontrar la transformación entre el modelo y la escena
- Para simplificar, vamos a ver cómo se puede obtener la transformación 2D-2D afín:

$$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} m_1 & m_2 \\ m_3 & m_4 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \end{bmatrix}$$

donde las variables  $m$  son los parámetros de rotación y escala y los  $t$  son los de traslación



## Reconocimiento características: transformación

$$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} m_1 & m_2 \\ m_3 & m_4 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \end{bmatrix}$$

- Tenemos un sistema de varias ecuaciones con varias incógnitas, reescribimos la ecuación de arriba

$$\begin{bmatrix} x & y & 0 & 0 & 1 & 0 \\ 0 & 0 & x & y & 0 & 1 \\ \dots & & \dots & & & \end{bmatrix} \begin{bmatrix} m_1 \\ m_2 \\ m_3 \\ m_4 \\ t_x \\ t_y \end{bmatrix} = \begin{bmatrix} u \\ v \\ \vdots \end{bmatrix}$$

- Cada emparejamiento introduce dos nuevas filas a la primera y última matriz. Nombramos las matrices como:

$$\mathbf{Ax} = \mathbf{b}$$



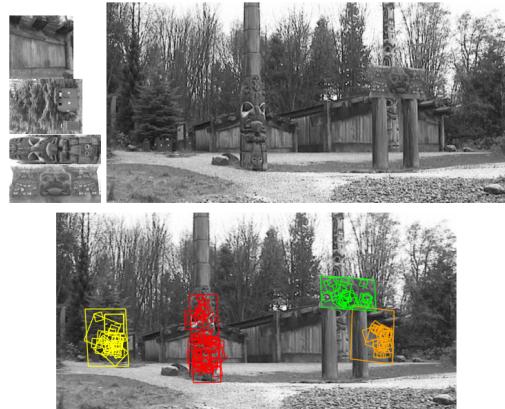
## Reconocimiento características: transformación

Resolvemos el sistema anterior mediante mínimos cuadrados

Consiste en encontrar la matriz  $x$  que minimiza el error cuadrático medio entre todos los emparejamientos

- Se resuelve este sistema:

$$\mathbf{x} = [\mathbf{A}^T \mathbf{A}]^{-1} \mathbf{A}^T \mathbf{b},$$



12 Extracción de características, segmentación y reconocimiento de objetos

45

## Reconocimiento de caras

- Algoritmo de Viola&Jones: Robust Real-time Object Detection. International Journal of Computer Vision. 2001
- El objetivo es detectar caras, no reconocerlas
- Otro objetivo es que sea muy rápido: este método tarda pocos milisegundos en procesar una imagen
- El método tiene pocos falsos positivos y un alto porcentaje de detección correcta
- Sólo sirve para caras frontales o con poco giro
- Le afecta el cambio de luminosidad

12 Extracción de características, segmentación y reconocimiento de objetos

46

## Estructura general del método

- Extracción de características
  - Uso de imagen integral!!
  - Extracción de muchísimas características
- Selección de características
  - Algoritmo AdaBoost de entrenamiento
- Cascada de clasificadores
  - Conseguir más velocidad

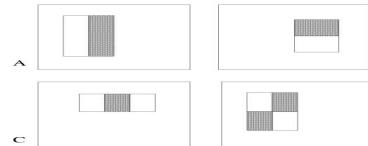
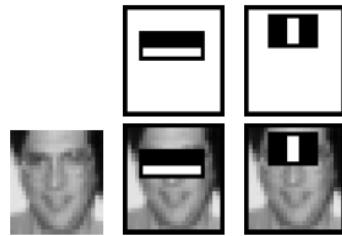
## Extracción de características

- Las caras comparten algunas características comunes:
  - La región del puente de la nariz es más clara que la de los ojos
  - La región de los pómulos es también más clara que los ojos
- Características a buscar:
  - Localizar nariz-ojos
  - Valores: claro-oscuro



## Características en rectángulo

- Valor de la característica
  - $\sum(\text{píxeles en área negra}) - \sum(\text{píxeles en área blanca})$
- Tres tipos: dos, tres, cuatro-rectángulos
- Cada característica está relacionada con una localización especial en la sub-ventana
- Cada característica puede tener cualquier tamaño y orientación



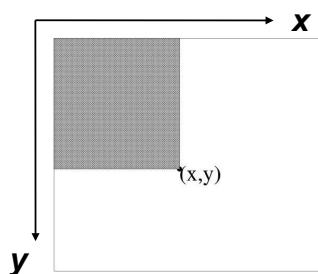
12 Extracción de características, segmentación y reconocimiento de objetos

49

## Imagen Integral

- Dada una resolución de ventana de 24x24 existen 180000 posibles características!
- Por ello, es necesario un cálculo rápido
- Introducen el uso de la imagen integral: cada punto en la imagen integral es la suma de todos los píxeles de la imagen original a la izquierda y arriba de ese punto

$$\begin{aligned} \text{imagen original} &= I(x, y) \\ \text{imagen integral} &= II(x, y) = \sum_{x' \leq x, y' \leq y} I(x', y') \end{aligned}$$



12 Extracción de características, segmentación y reconocimiento de objetos

50

## Cálculo de la imagen Integral

- Se puede calcular en un solo paso:

$$\begin{aligned}s(x, y) &= s(x, y-1) + i(x, y) \\ ii(x, y) &= ii(x-1, y) + s(x, y)\end{aligned}$$

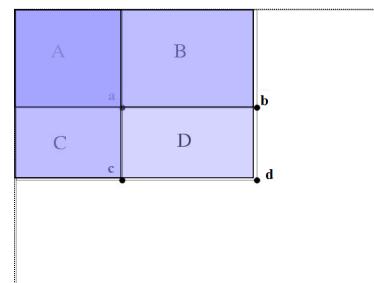
 donde  $s(x, y)$  es la suma acumulada de la columna. Hay que tener en cuenta las primeras fila y columna

Imagen	Imagen integral
--------	-----------------

0	1	1	1		0	1	2	3
1	2	2	3		1	4	7	11
1	2	1	1		2	7	11	16
1	3	1	0		3	11	16	21

## ¿Qué ventaja tiene la imagen integral?

- Como lo que queremos calcular es la suma de los píxeles dentro de un rectángulo, la imagen integral permite hacer este cálculo con tres operaciones básicas
- Para calcular características con dos, tres o cuatro rectángulos es necesario realizar 6, 8 y 9 referencias a la imagen, respectivamente (ejemplo con 2)

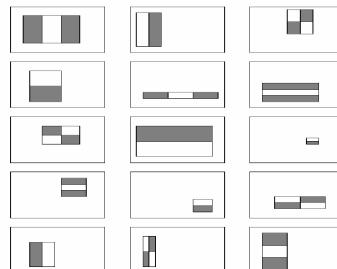


$$\begin{aligned}ii(a) &= A \\ ii(b) &= A+B \\ ii(c) &= A+C \\ ii(d) &= A+B+C+D \\ D &= ii(d)+ii(a)-ii(b)-ii(c)\end{aligned}$$

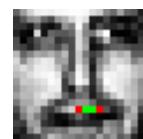


## Selección de características

- Demasiadas características: en una ventana 24x24 hay 180000 posibles combinaciones de tamaño y orientación
- Hay que seleccionar aquellas que sean relevantes para detectar una cara
- Para ello, podemos usar el algoritmo AdaBoost y The Attentional Cascade



Carac. relevante



Carac. irrelevante

## Referencias

- Lowe, D. G., "Distinctive Image Features from Scale-Invariant Keypoints", International Journal of Computer Vision, 60, 2, pp. 91-110, 2004.
- Duda, R. O. and P. E. Hart, "Use of the Hough Transformation to Detect Lines and Curves in Pictures," Comm. ACM, Vol. 15, pp. 11-15 (January, 1972)
- Transformada de Hough: [http://en.wikipedia.org/wiki/Hough\\_transform](http://en.wikipedia.org/wiki/Hough_transform)
- K-medias: [http://en.wikipedia.org/wiki/K-means\\_clustering](http://en.wikipedia.org/wiki/K-means_clustering)
- Método de Viola&Jones para reconocimiento de caras:  
[http://research.microsoft.com/~viola/Pubs/Detect/violaJones\\_IJCV.pdf](http://research.microsoft.com/~viola/Pubs/Detect/violaJones_IJCV.pdf)
- <http://www.pigeon.psy.tufts.edu/avc/kirkpatrick/default.htm>