



Visión Artificial

Tema 3: Detección de características

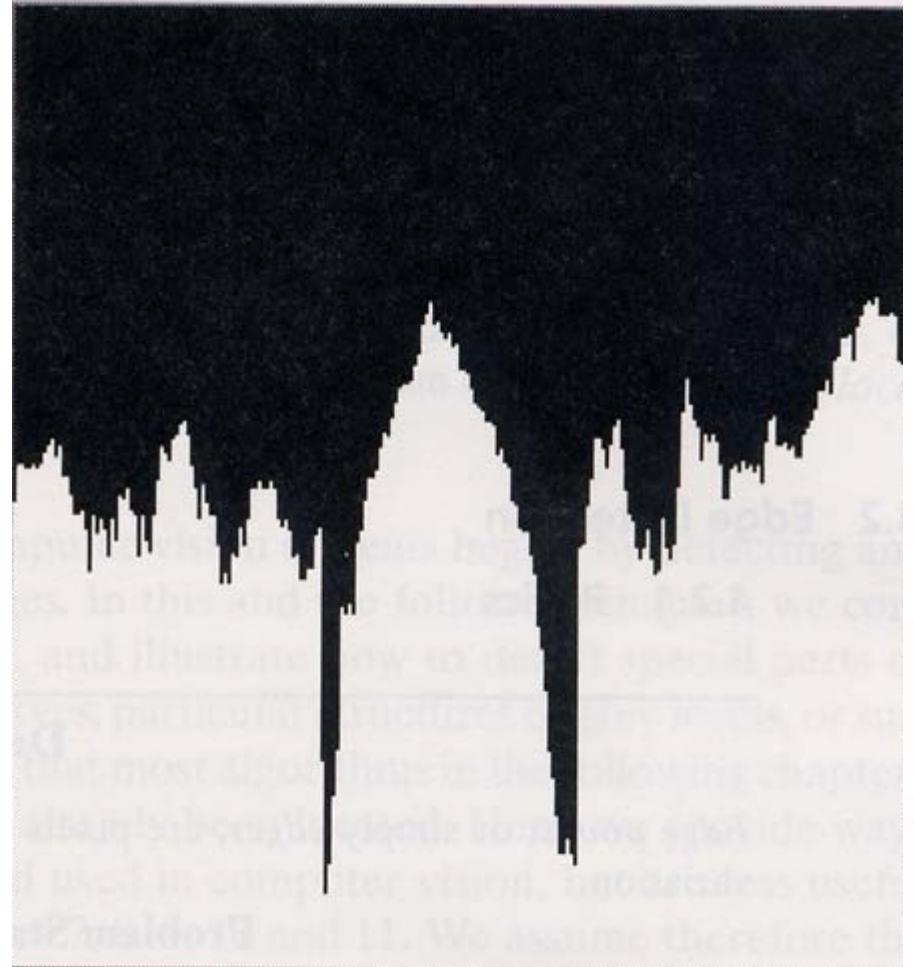
- ▶ Bordes
- ▶ Esquinas
- ▶ Puntos característicos
- ▶ Líneas y curvas

Bordes

- ▼ Detección de bordes
- ▼ El método de Canny

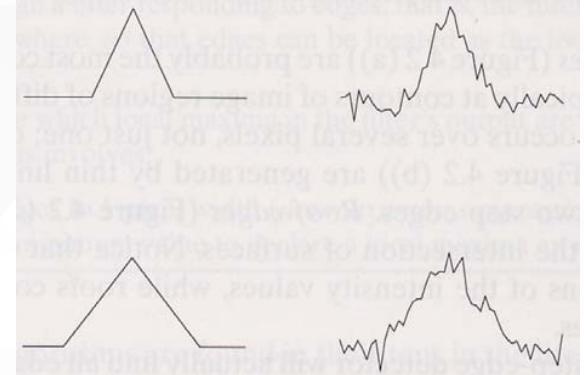
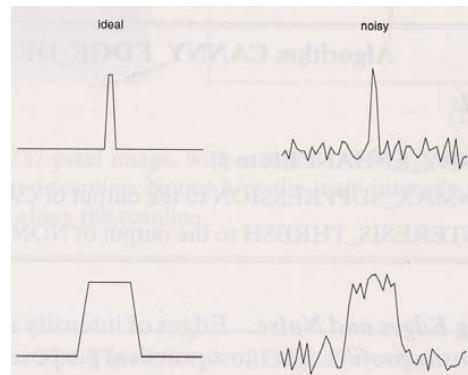
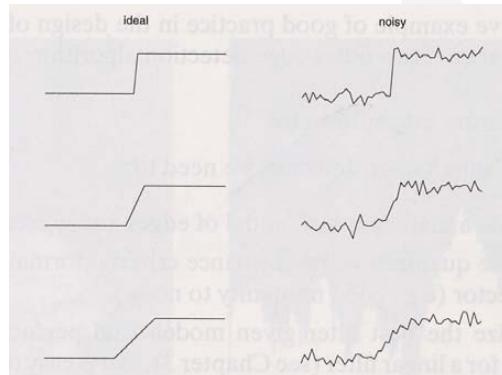
Detección de bordes

- Se corresponden con aquellos píxeles en los que la función de intensidad varía bruscamente en su entorno local.



Detección de bordes

- ▼ Tipos de bordes:
 - ▼ Escalón: contornos que delimitan regiones de diferentes intensidades.
 - ▼ Cresta: se generan por líneas finas. Las líneas gruesas se identifican con un patrón de tipo escalón.
 - ▼ Tejado: pueden aparecer en la intersección de superficies, aunque son relativamente infrecuentes.



Detección de bordes

- ▼ Pasos fundamentales en la detección de bordes:
 - ▼ Suavizado:
 - ▼ Eliminar el ruido de la imagen sin perder la información de los bordes reales.
 - ▼ Realce de bordes:
 - ▼ Aplicar un filtro que proporcione respuestas altas a los puntos que sigan un patrón de borde.
 - ▼ Localización de bordes:
 - ▼ Distinguir entre respuestas altas del filtro originadas por zonas ruidosas de los bordes reales:
 - ▼ determinar los máximos locales.
 - ▼ establecer un valor mínimo para considerar como borde un máximo local.

El método de Canny

- ▼ Resumen del método de Canny:
 - ▼ Dada una imagen I :
 - ▼ Aplicar *realce de bordes* a I : resultado en I_E
 - ▼ Suavizado y cálculo del gradiente
 - ▼ Aplicar *supresión del no máximo* a I_E : resultado en I_M
 - ▼ Encontrar los máximos locales de I_E en la dirección normal al borde. Suprimir los restantes pixels
 - ▼ Aplicar *umbralización por histéresis* a I_M : resultado en I_H
 - ▼ Eliminar los píxels de borde originados por el ruido de la imagen
 - ▼ La imagen de bordes se obtiene en I_H .

El método de Canny: realce de bordes

- ▼ Aplicar un suavizado gaussiano a I para obtener J : $J = G^*I$
- ▼ Por cada píxel (i, j) :
 - ▼ Calcular las componentes de gradiente (J_x, J_y)
 - ▼ Estimar la intensidad del borde como:

$$e_s(i, j) = \sqrt{J_x^2 + J_y^2}$$

- ▼ Estimar la normal del borde:

$$e_o(i, j) = \arctan \frac{J_y}{J_x}$$

- ▼ La salida es la imagen de intensidad $e_s(i, j)$ y la imagen de orientación $e_o(i, j)$

El método de Canny: realce de bordes

- ▼ Cálculo de las componentes del gradiente
 - ▼ Dada una función 2D $f(x,y)$, su derivada parcial, con respecto a x , se define como:

$$\frac{\delta f(x, y)}{\delta x} = \lim_{\varepsilon \rightarrow 0} \frac{f(x + \varepsilon, y) - f(x, y)}{\varepsilon}$$

- ▼ Para datos discretos, se puede aproximar utilizando diferencias finitas:

$$\frac{\delta f(x, y)}{\delta x} = \frac{f(x+1, y) - f(x, y)}{1}$$

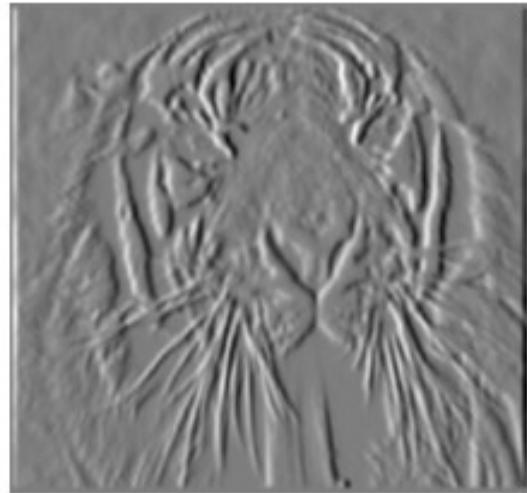
- ▼ ¿Cuál es filtro asociado para implementar la expresión anterior a través de una convolución?

El método de Canny: realce de bordes

$$\frac{\partial f(x, y)}{\partial x}$$

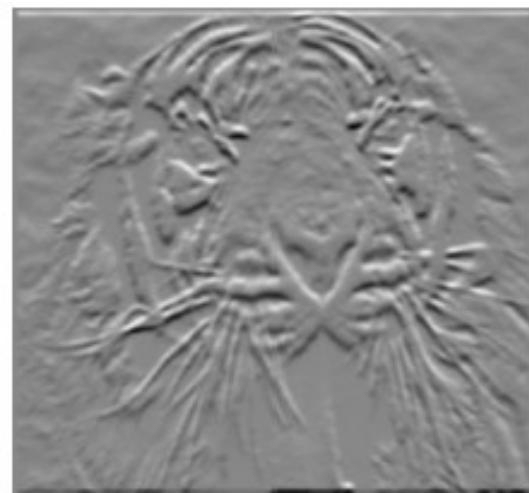
$$\begin{bmatrix} -1 & 1 \\ 1 & -1 \end{bmatrix}$$

$$\begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}$$



$$\frac{\partial f(x, y)}{\partial y}$$

$$\begin{bmatrix} -1 & 1 \\ 1 & -1 \end{bmatrix}$$



El método de Canny: realce de bordes

▼ Otras aproximaciones de filtros de gradiente:

Prewitt

D_x
-1 0 1
-1 0 1
-1 0 1

D_y
-1 -1 -1
0 0 0
1 1 1

Cálculo de gradiente con suavizado media

Sobel

-1 0 1
-2 0 2
-1 0 1

-1 -2 -1
0 0 0
1 2 1

Cálculo de gradiente con aproximación a suavizado gaussiano

Roberts

0 1
-1 0

1 0
0 -1

Respuestas altas ante bordes diagonales (sensible al ruido)

El método de Canny: supresión del no máximo

- ▼ El resultado de aplicar el paso de *realce de bordes* proporciona zonas de alta intensidad de borde alrededor de máximos locales. La fase de *supresión del no máximo* produce bordes con anchura de 1 píxel.
- ▼ Algoritmo *Supresión del no máximo*
 - ▼ Por cada píxel (i, j) :
 - ▼ Considerando 4 posibles orientaciones ($0^\circ, 45^\circ, 90^\circ, 135^\circ$), calcular la orientación d_k que mejor aproxima $e_o(i, j)$.
 - ▼ Si $e_s(i, j)$ es menor que al menos uno de sus dos vecinos en la orientación d_k , asignar $I_M(i, j) = 0$ (suprimir el borde).
 - ▼ Si no, asignar $I_M(i, j) = e_s(i, j)$.
 - ▼ El resultado es una imagen de puntos de borde $I_M(i, j)$.

El método de Canny: supresión del no máximo

Resultados de realce y supresión del no máximo con distintos filtros gaussianos



$\sigma = 1$



$\sigma = 2$



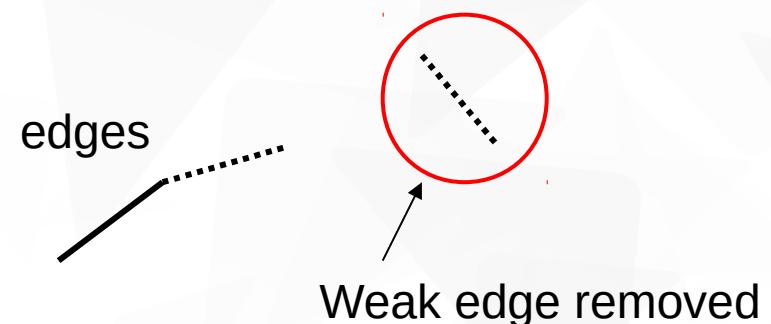
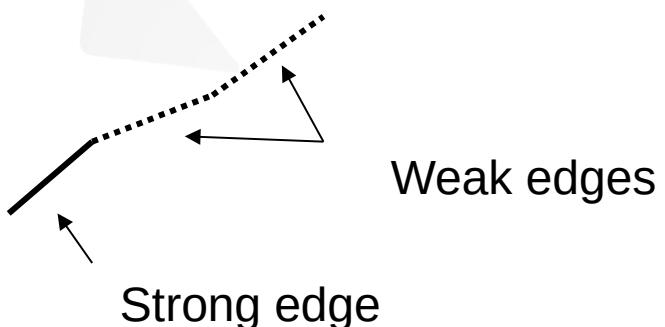
$\sigma = 3$

El método de Canny: umbralización por histéresis

- Objetivo: descartar los máximos locales generados por el ruido de la imagen.
- Suposiciones:
 - En general, los bordes reales deberían tener una intensidad alta.
 - Los puntos pertenecientes a bordes reales están conectados con otros puntos de borde.
- Una posibilidad sería descartar aquellos puntos cuyo valor de borde $I_M(i, j)$ no supere un cierto umbral:
 - Problema: algunos puntos de borde (real) tienen menor intensidad que puntos falsos de borde.
 - Solución: umbralización por histéresis – un punto pertenece a un borde real si su intensidad de borde es de al menos τ_l y está enlazado a algún punto cuya intensidad sea mayor que τ_h , siendo $\tau_h > \tau_l$

El método de Canny: umbralización por histéresis

- ▼ Algoritmo *umbralización por histéresis*
- ▼ Para todos los puntos de I_M , recorriendo la imagen en un orden fijo:
 - ▼ Localizar el siguiente píxel de I_M no visitado, tal que $I_M(i,j) > \tau_h$
 - ▼ Comenzando por $I_M(i, j)$, seguir la cadena de máximos locales conectados, en las dos direcciones perpendiculares a la normal del borde, mientras $I_M > \tau_l$. Marcar todos los puntos como visitados (Imagen I_H)
- ▼ La salida es una imagen binaria (I_H) que representa la situación de pertenencia de cada píxel a un borde.



El método de Canny: umbralización por histéresis

Resultados finales del algoritmo de Canny con distintos filtros gaussianos



$$\sigma = 1$$



$$\sigma = 2$$



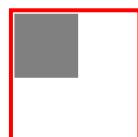
$$\sigma = 3$$

Esquinas

- ▼ Definición de esquina
- ▼ Detección de esquinas

Definición de esquina

- ▶ Puntos donde coinciden dos bordes: alto gradiente en dos direcciones.
- ▶ La cualidad de esquina no está definida en un único punto dado que sólo existe una dirección de gradiente por punto.
- ▶ Es necesario considerar el gradiente en el entorno local de cada píxel.
- ▶ Categorías de ventanas de imagen en base al gradiente:
 - ▶ Constante (homogénea): poca o ninguna variación de intensidad.
 - ▶ Borde: fuerte cambio de intensidad en una dirección.
 - ▶ Esquina: Fuerte variación de intensidad en direcciones ortogonales.



Detección de esquinas

- Sean E_x y E_y las imágenes de gradiente horizontal y vertical, la detección de esquinas supone localizar aquellos puntos en cuyo entorno E_x y E_y presentan una magnitud alta.
- No es posible fijar únicamente un umbral, dado que ambas componentes de gradiente presentan valores altos en bordes diagonales.
- Solución: analizar la distribución de gradientes en el entorno del píxel.
 - Por cada píxel p , utilizar su entorno local para formar la siguiente matriz:
$$C = \begin{bmatrix} \sum E_x^2 & \sum E_x E_y \\ \sum E_x E_y & \sum E_y^2 \end{bmatrix}$$
 - Si el menor autovalor de esta matriz es superior a un cierto umbral, considerar p esquina.

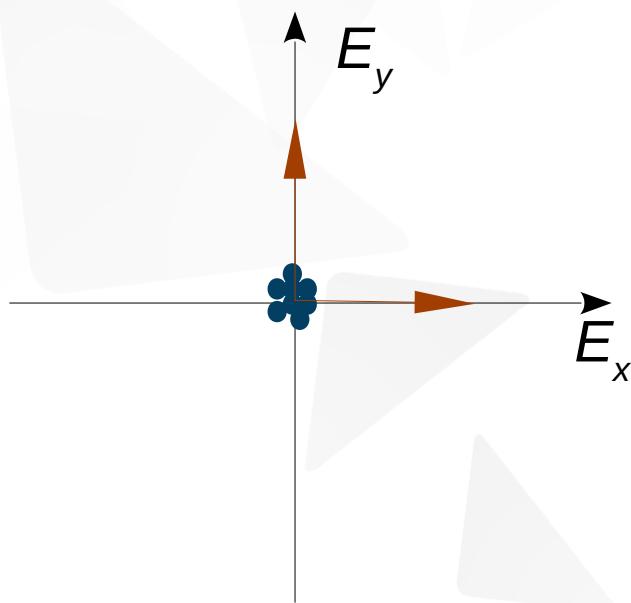
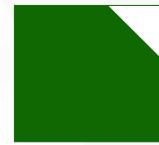
Detección de esquinas

- C es la matriz de covarianza (autocorrelación) de los vectores de gradiente en el entorno del píxel.
- Puesto que C es simétrica, puede diagonalizarse aplicando una rotación R , siendo λ_1 y λ_2 los autovalores de C :

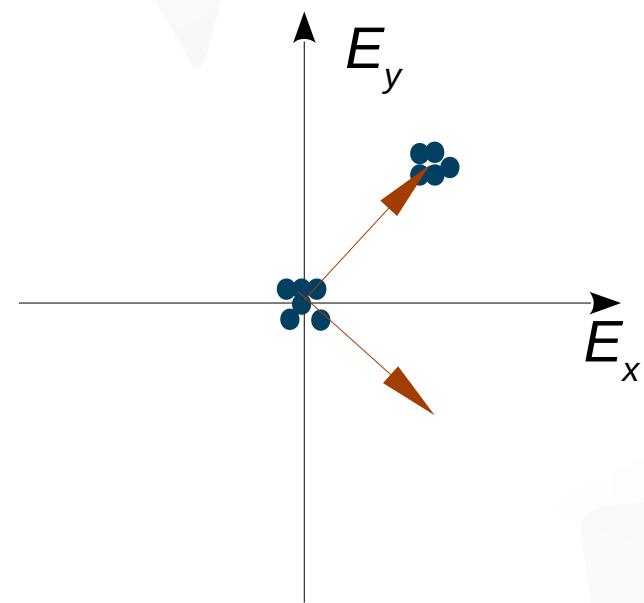
$$C = R^T \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} R$$

- La matriz diagonal representa las variaciones de gradiente en los ejes definidos por R .
- Se pueden distinguir tres casos ideales:
 - Región homogénea: $\lambda_1 = \lambda_2 = 0$
 - Borde: $\lambda_1 > 0, \lambda_2 = 0$
 - Esquina: $\lambda_1 \geq \lambda_2 > 0$

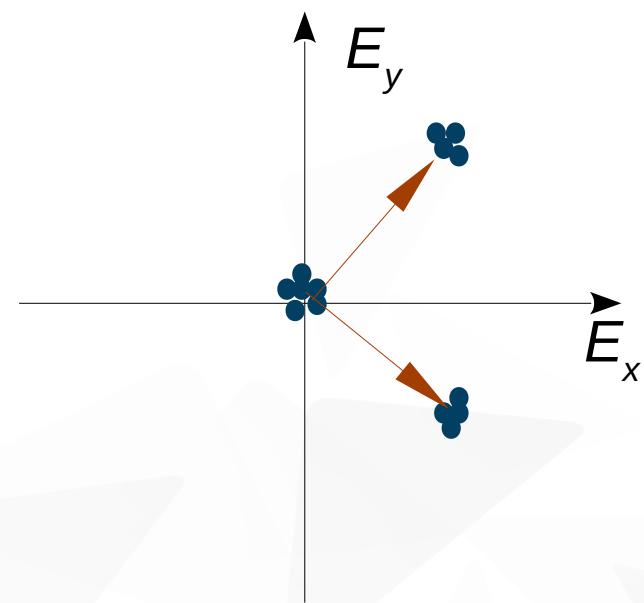
Detección de esquinas



región homogénea



borde



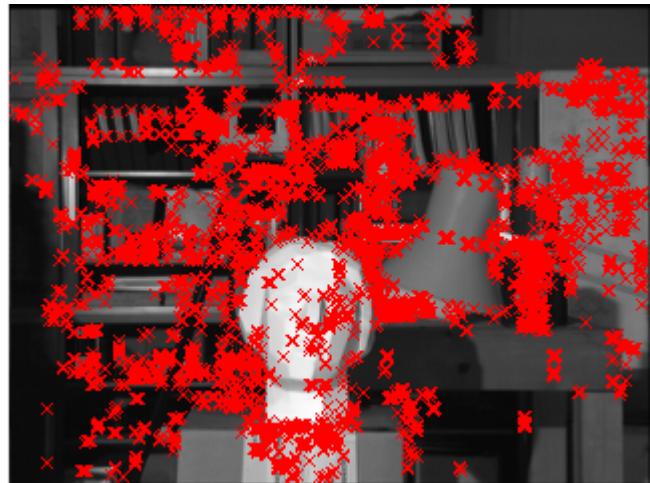
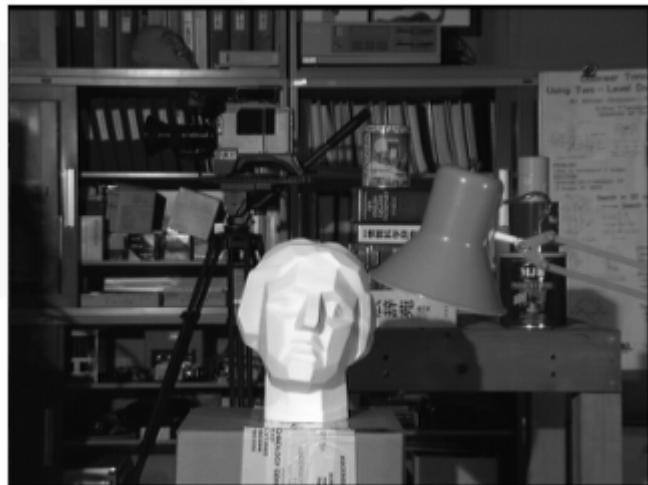
esquina

Detección de esquinas

▼ Algoritmo: *detección*

- ▼ Calcular las imágenes de gradiente.
- ▼ Por cada píxel p :
 - ▼ Construir la matriz C utilizando un entorno local de p de tamaño $(2N+1) \times (2N+1)$.
 - ▼ Calcular λ_2 , el menor autovalor de C .
 - ▼ Si $\lambda_2 > \tau$, almacenar las coordenadas de p en una lista L .
- ▼ Ordenar L en orden decreciente de λ_2
- ▼ Recorrer la lista L partiendo del primer elemento: por cada punto p de L , borrar todos los puntos que aparezcan a continuación en la lista y que pertenezcan al entorno de p
- ▼ El resultado es una lista de puntos esquina que no se solapan.

Detección de esquinas

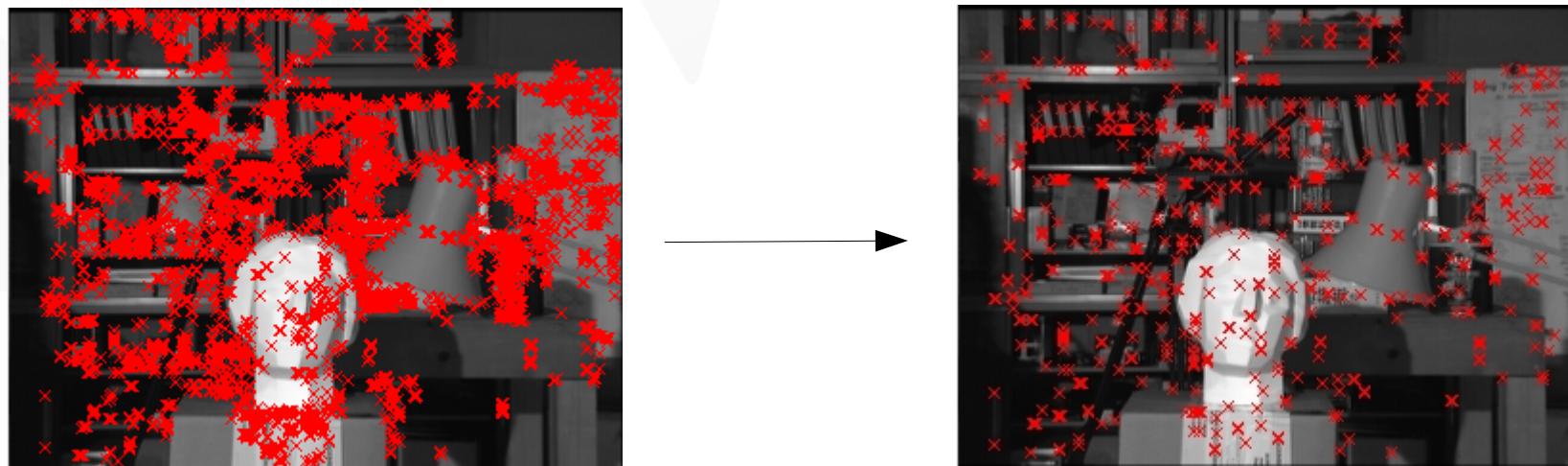


Detección de esquinas

▼ Algoritmo: supresión del no máximo

- ▼ Calcular las imágenes de gradiente.
- ▼ Por cada píxel p :
 - ▼ Construir la matriz C utilizando un entorno local de p de tamaño $(2N+1) \times (2N+1)$.
 - ▼ Calcular λ_2 , el menor autovalor de C .
 - ▼ Si $\lambda_2 > \tau$, almacenar las coordenadas de p en una lista L .
- ▼ Ordenar L en orden decreciente de λ_2
- ▼ Recorrer la lista L partiendo del primer elemento: por cada punto p de L , borrar todos los puntos que aparezcan a continuación en la lista y que pertenezcan al entorno de p
- ▼ El resultado es una lista de puntos esquina que no se solapan.

Detección de esquinas



Detección de esquinas

- ▼ Medidas de esquina: *análisis de la matriz de covarianza (C)*
 - ▼ *Shi & Tomasi:*
 - ▼ $m_c = \text{mínimo autovalor de } C$
 - ▼ Estabilidad frente a otros detectores.
 - ▼ *Harris & Stephens:*
 - ▼ $m_c = \det(C) - k * \text{traza}^2(C) = \lambda_1 \lambda_2 - k * (\lambda_1 + \lambda_2)^2$
 - ▼ k es una constante en el rango [0.04, 0.15]
 - ▼ Mayor velocidad de cálculo que el método anterior.
 - ▼ *Noble:*
 - ▼ $m_c = \det(C) / \text{traza}(C)$
 - ▼ Evita el uso de un parámetro adicional (k).

Puntos característicos

- ▶ Detección
- ▶ Descriptores
- ▶ Búsqueda de correspondencias

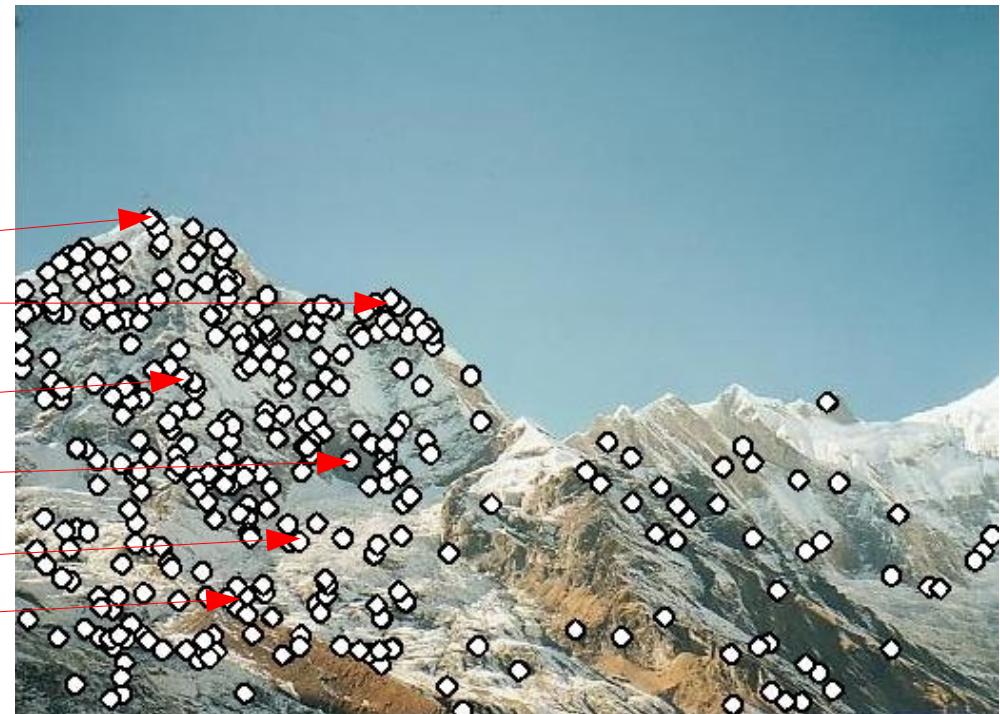
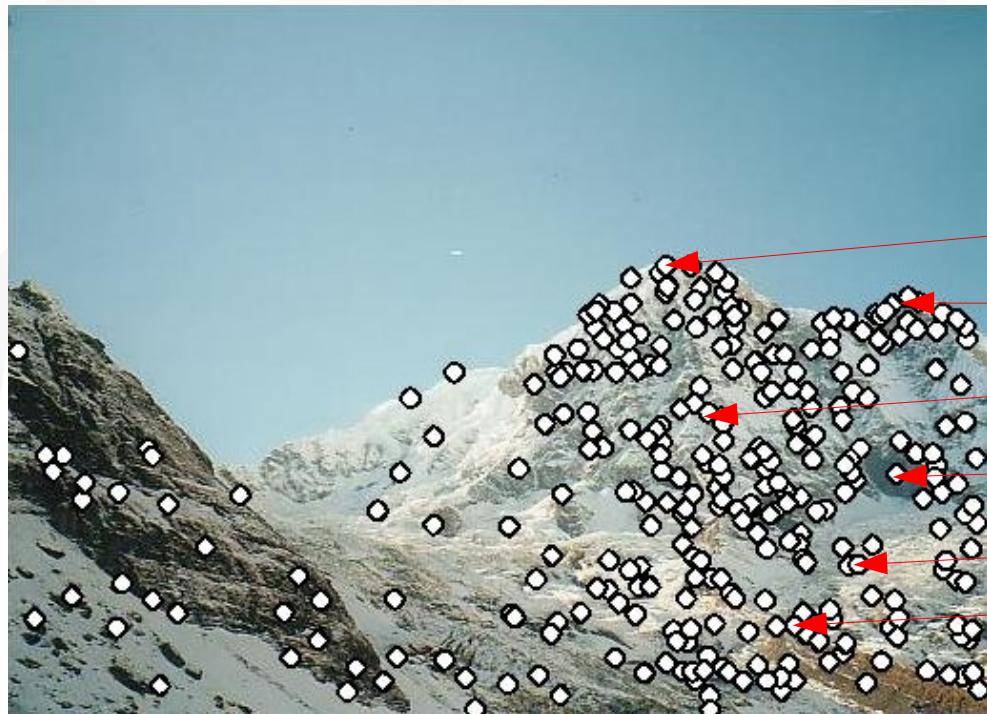
Puntos característicos

Creación de panoramas: ¿cómo combinar las dos imágenes para crear un panorama?



Puntos característicos

Extraer características comunes y ponerlas en correspondencia



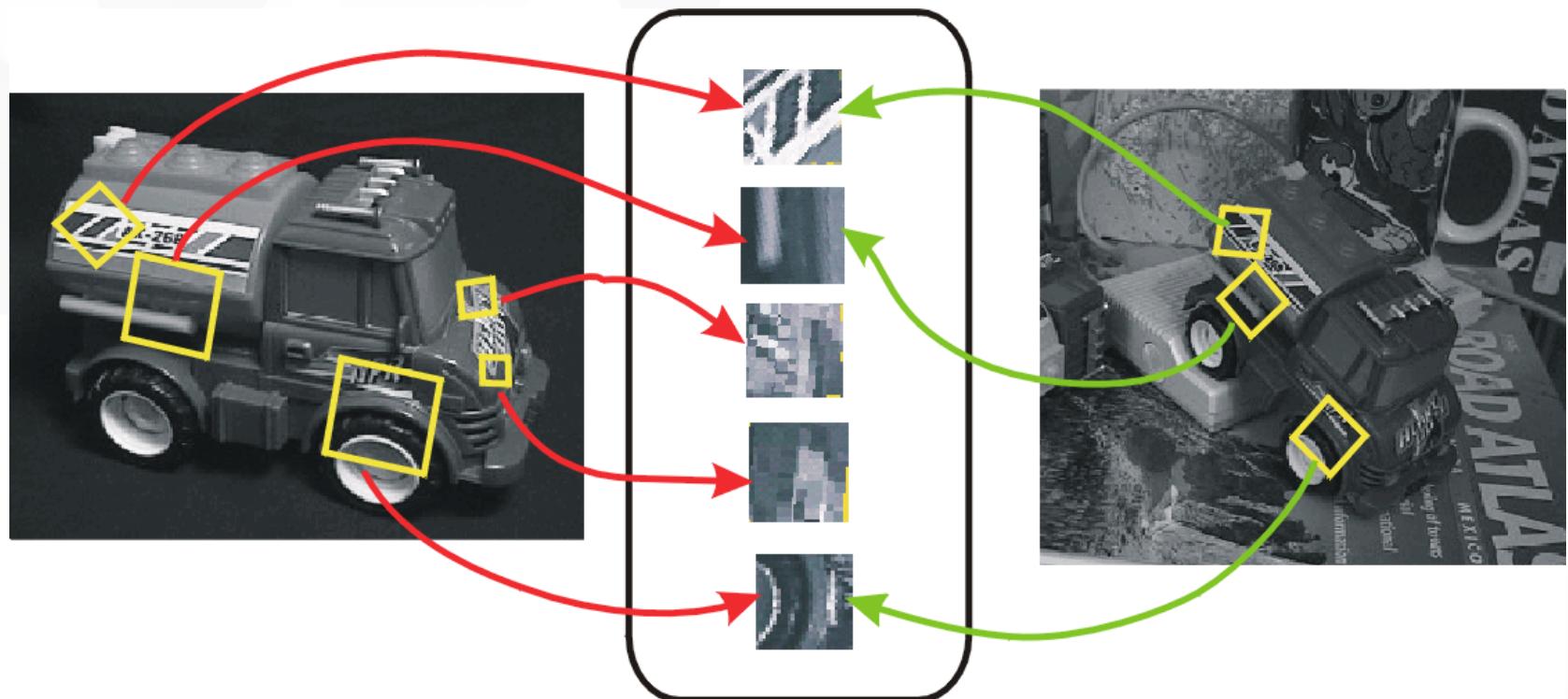
Puntos característicos

Alinear ambas imágenes a partir de las correspondencias obtenidas



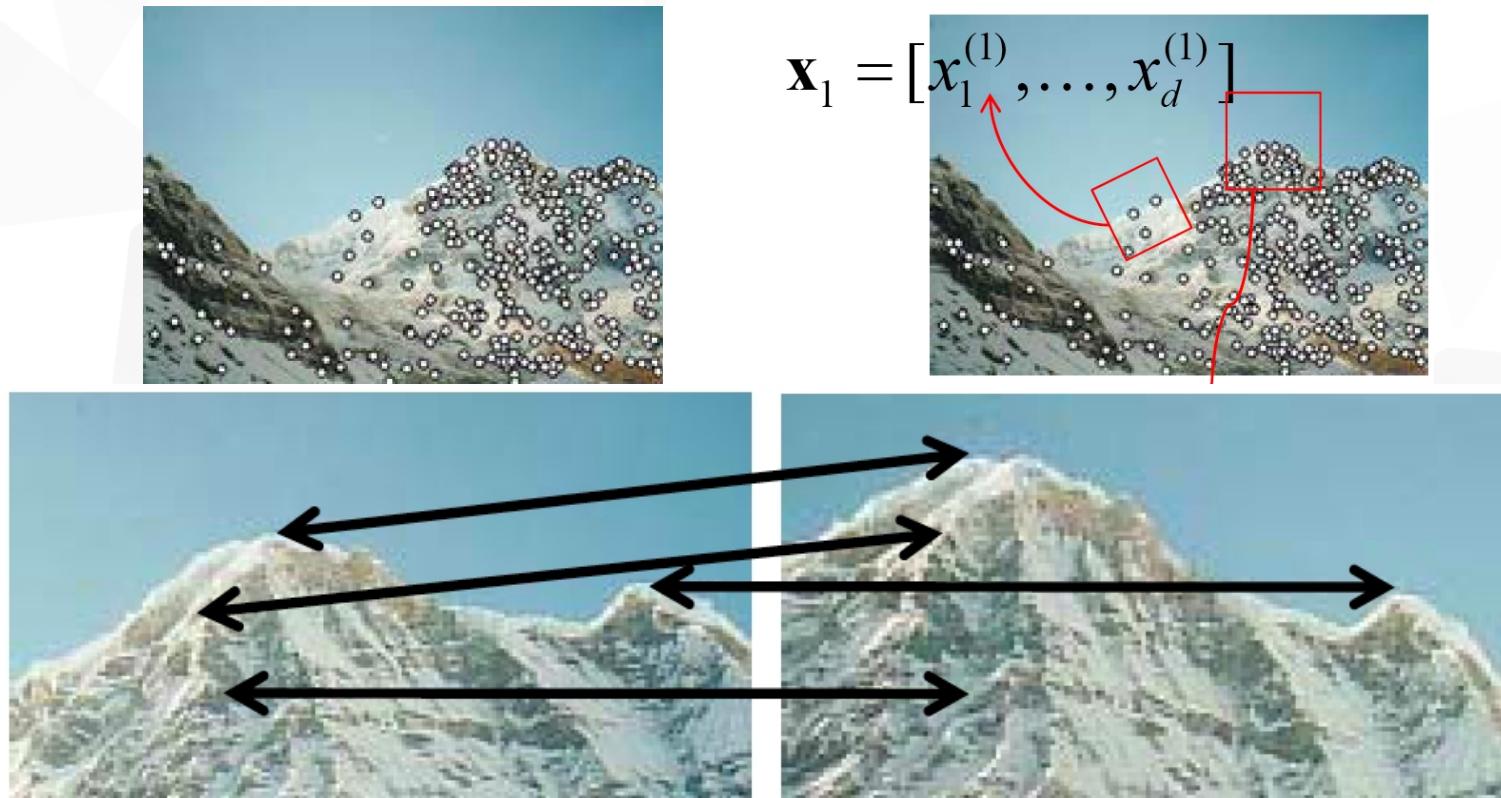
Puntos característicos

Otra aplicación: reconocimiento de objetos



Puntos característicos

- ▼ **Detección:** identificar puntos de interés.
- ▼ **Descripción:** extraer un vector descriptor alrededor de cada punto.
- ▼ **Correspondencia:** buscar correspondencias entre descriptores.



Puntos característicos: detección

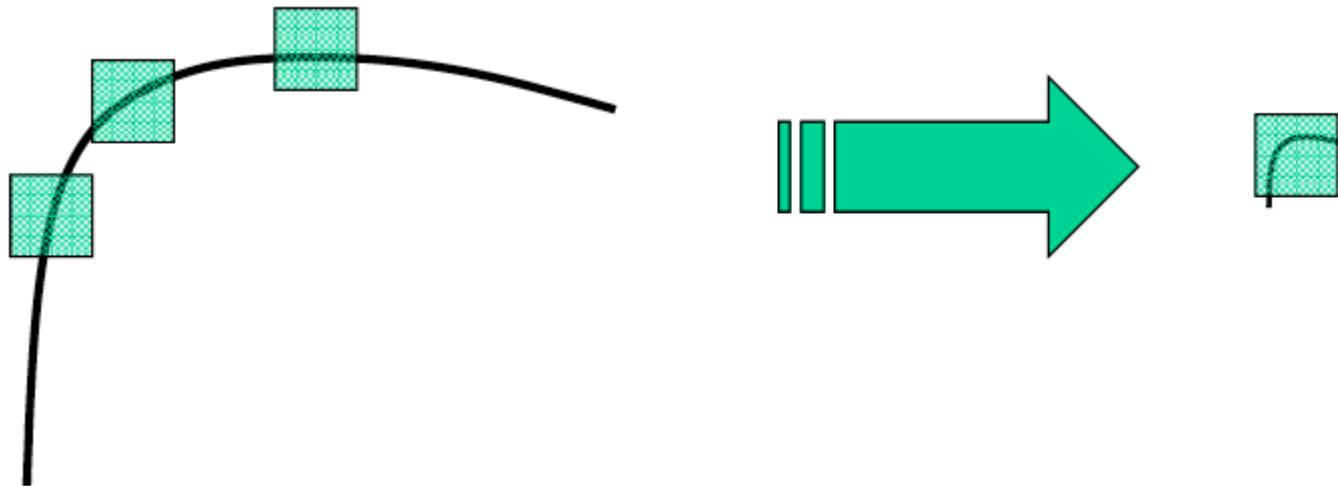
- ▶ **¿Qué puntos escoger?**
- ▶ Las zonas sin textura son imposibles de localizar.
- ▶ Las zonas con fuertes cambios de contraste (alto gradiente) pueden localizarse más fácilmente.
- ▶ **¿Puntos de borde?**
 - ▶ Problema de la apertura: sólo es posible poner en correspondencias dos zonas de borde en la dirección normal a la dirección del borde.
- ▶ Para evitar ambigüedades, es necesario que exista gradiente en dos direcciones: **ESQUINAS (Harris)** y **MANCHAS (Hessian)**

Puntos característicos: detección

- ▶ Para que la búsqueda de correspondencias sea robusta, es necesario que las características sean invariantes a:
 - ▶ Transformaciones geométricas: rotaciones, traslaciones, cambios de escala.
 - ▶ Transformaciones fotométricas: cambios de iluminación.
- ▶ Propiedades de los detectores de esquinas
 - ▶ Invarianza a cambios de iluminación: métodos basados en el análisis del gradiente en el entorno de cada píxel.
 - ▶ Invarianza a rotaciones: al rotar el entorno de un píxel, los autovalores de la matriz de covarianza mantienen su valor.
 - ▶ ¿Invarianza a cambios de escala?

Puntos característicos: detección

▼ Detección de esquinas en diferentes escalas

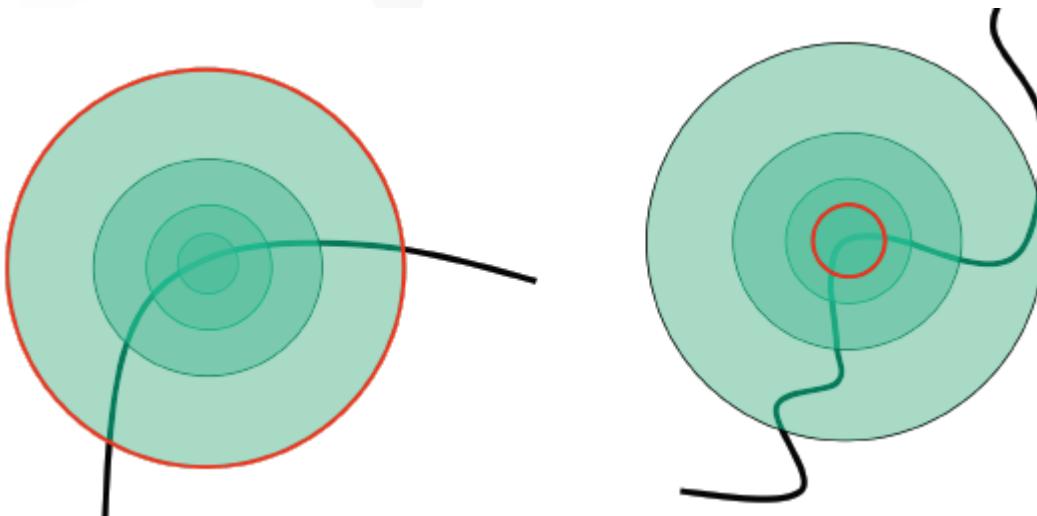


*Todos los píxeles se
consideran puntos
de borde*

*En una escala
diferente: **esquina***

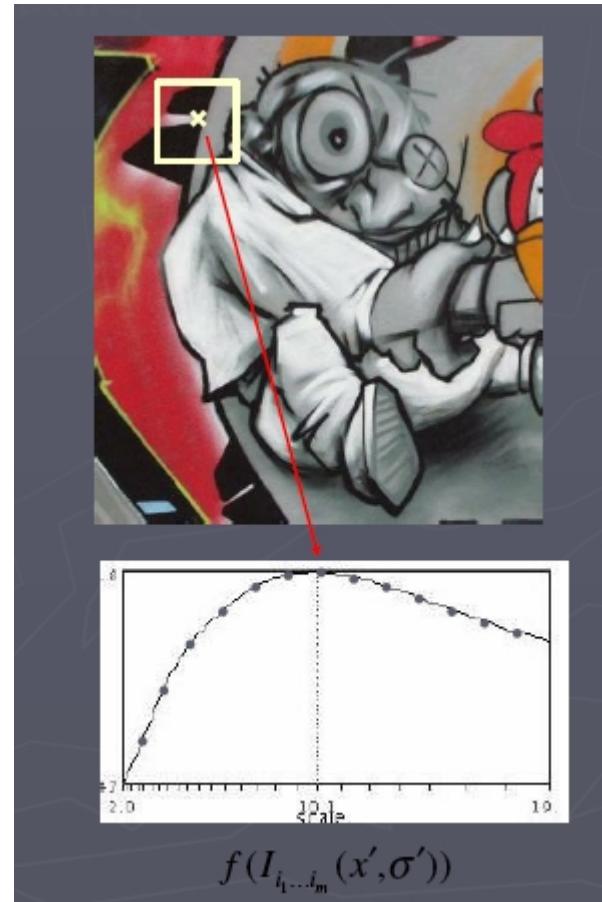
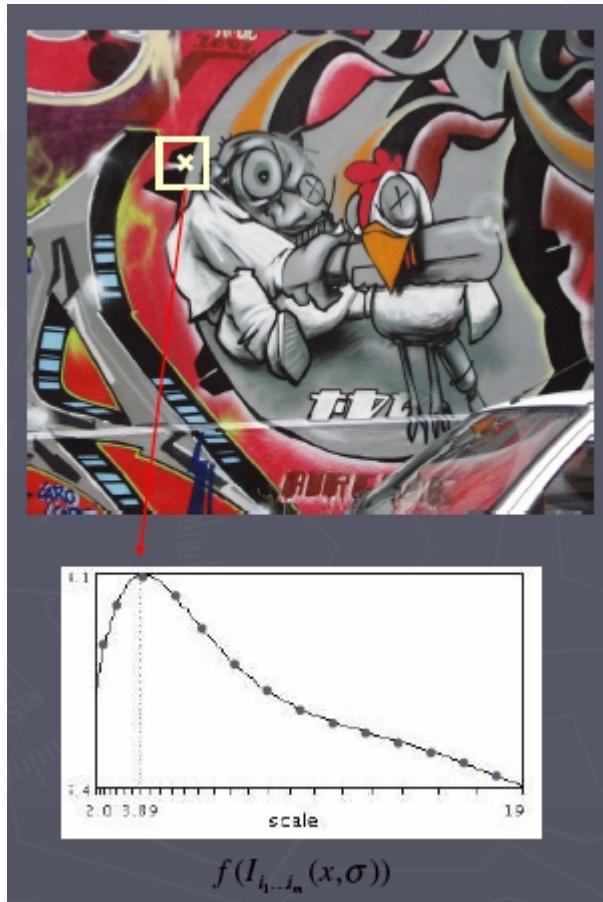
Puntos característicos: detección

- ▼ Invarianza a los cambios de escala: búsqueda en entornos con diferentes tamaños



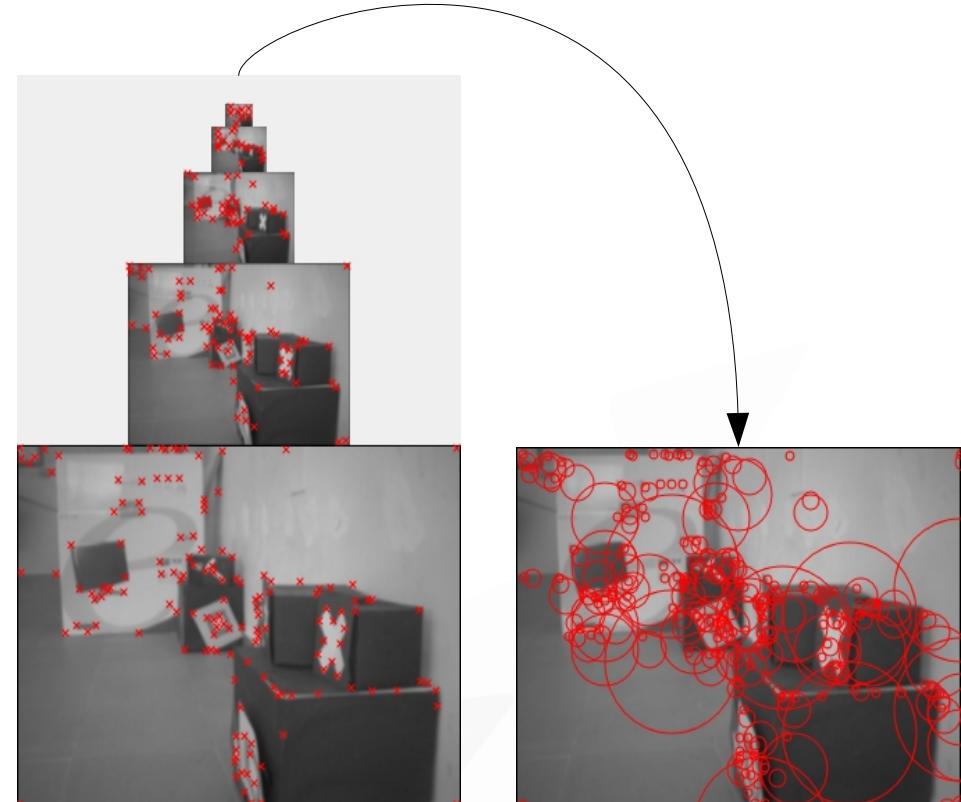
Puntos característicos: detección

- ▼ Invarianza a los cambios de escala: búsqueda en la imagen con diferentes resoluciones.



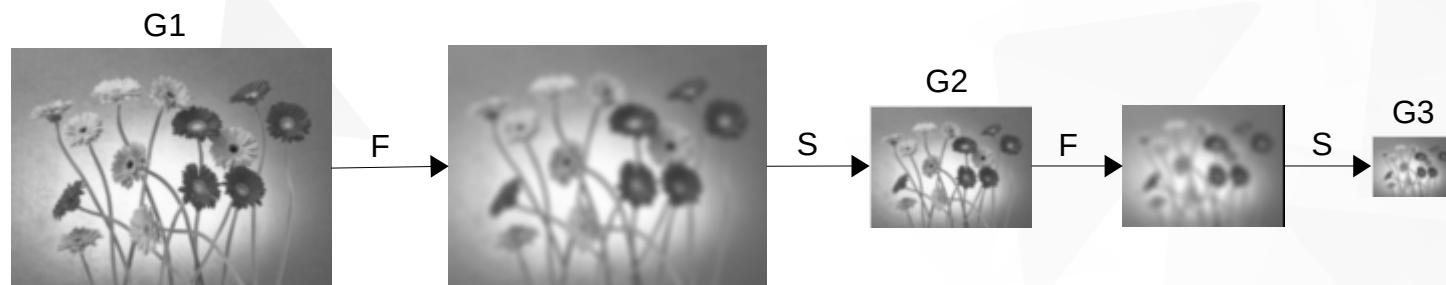
Puntos característicos: detección

- ▼ Espacio de escala de una imagen
 - ▼ Estructura piramidal.
 - ▼ Base de la pirámide: imagen original.
 - ▼ Cada nivel se construye reduciendo la resolución del anterior.
 - ▼ En la imagen original, cada característica se trata como una región del tamaño que corresponda.



Puntos característicos: detección

- ▼ No es suficiente con submuestrear la imagen: pérdida de información relevante.
- ▼ Solución: filtrado paso bajo + submuestreo
- ▼ Filtro paso bajo: enfatiza las bajas frecuencias suavizando la imagen.
- ▼ Características del filtro: normalizado, simétrico, decreciente hacia los extremos.
- ▼ Filtro gaussiano: pirámide gaussiana.



Puntos característicos: detección

▼ Detectores de puntos característicos

▼ Detector Sift (Scale Invariant Feature Transform)

- ▼ Regiones tipo "mancha" en el espacio de escala.
- ▼ Por cada región se calcula su orientación principal (dirección principal del gradiente).

▼ Detector Surf (Speeded Up Robust Features)

- ▼ Inspirado en el detector Sift.
- ▼ Utilización de filtros cuadrados (reduce el coste computacional de aplicar una convolución).

▼ ORB (Oriented FAST and Rotated BRIEF)

- ▼ Esquinas FAST en el espacio de escala
- ▼ La orientación de la región se obtiene a partir del vector formado por el centro de la esquina y el centroide de intensidad

Puntos característicos: descriptores de región

- ▶ Descriptor: vector que describe una ventana centrada en un punto con un determinado radio.
- ▶ Propiedades: invarianza a rotación, tamaño, condiciones de iluminación, ...
- ▶ Uso: comparación de regiones.
- ▶ Alternativas:
 - ▶ Rift
 - ▶ Spin
 - ▶ **Sift**: histogramas de gradientes relativos a la orientación principal.
 - ▶ **Surf**: vector de gradientes horizontales y verticales según la orientación principal.
 - ▶ **ORB**: descriptores BRIEF (Binary Robust Independent Elementary Features) orientados.

Puntos característicos: descriptores de región

Descriptores Sift

- ▼ Utiliza un sistema local de coordenadas centrado en el centro de la región de interés.



- ▼ $\{x, y\}$: sistema global de la imagen.
- ▼ $\{u, v\}$: sistema local de la región.
- ▼ Se elige u apuntando hacia la orientación principal.
- ▼ El descriptor se calcula sobre la región en $\{u, v\}$.
- ▼ La caracterización sobre $\{u, v\}$ se mantiene invariante ante cambios de rotación, traslación, ...

Puntos característicos: descriptores de región

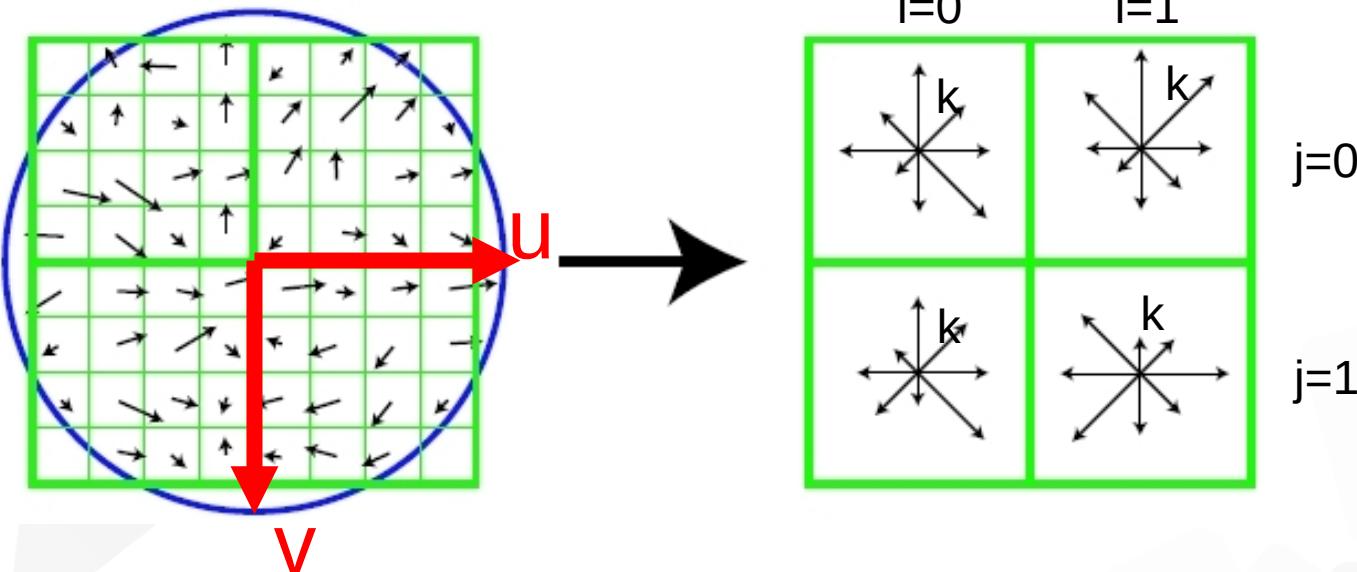
Descriptores Sift

▼ Generación del descriptor

- ▼ Se calcula el gradiente de la imagen.
- ▼ Se calcula el gradiente de la región de interés en el sistema local $\{u, v\}$.
- ▼ Se divide la región en $ni \times nj$ subregiones de tamaño $lu \times lv$.
- ▼ Por cada subregión, se genera un histograma del gradiente utilizando 8 direcciones principales.
- ▼ El descriptor es un vector formado por los distintos histogramas obtenidos.

Puntos característicos: descriptores de región

Descriptores Sift



- ▼ $n_i \times n_j = 2 \times 2$ subregiones = 4 subregiones
- ▼ $l_u \times l_v = 4 \times 4$ puntos por subregión
- ▼ 8 direcciones principales
- ▼ descriptor = 8 direcciones \times 4 subregiones = 32 elementos

Puntos característicos: descriptores de región

Descriptores Surf

▼ Generación del descriptor

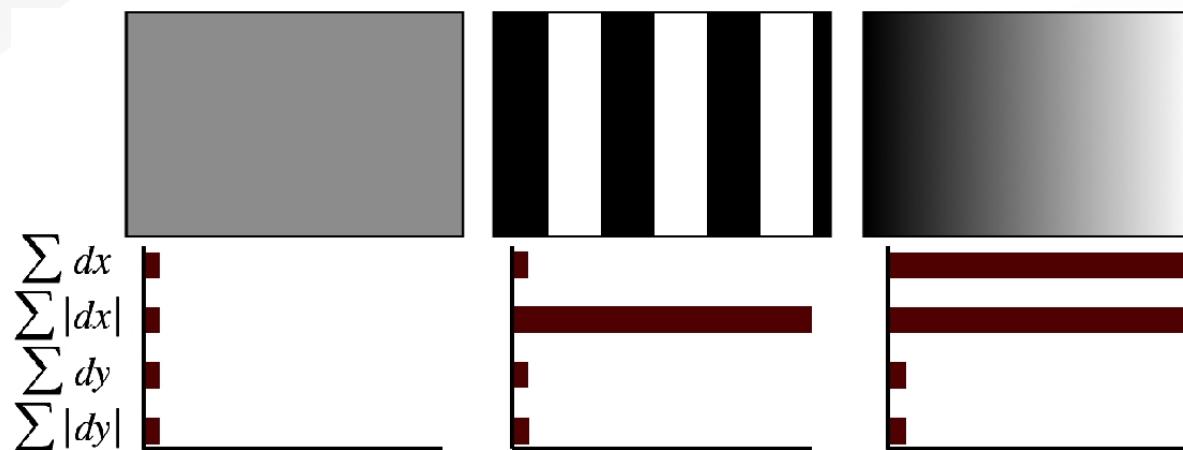
- ▼ El descriptor se calcula a partir de la región alineada con su orientación principal.
- ▼ Se usan 4x4 subregiones.
- ▼ Sólo se consideran los gradientes horizontales (d_x) y verticales (d_y).
- ▼ Por cada subregión, se obtiene el vector:

$$\mathbf{v} = (\sum d_x, \sum d_y, \sum |d_x|, \sum |d_y|)$$

Puntos característicos: descriptores de región

Descriptores Surf

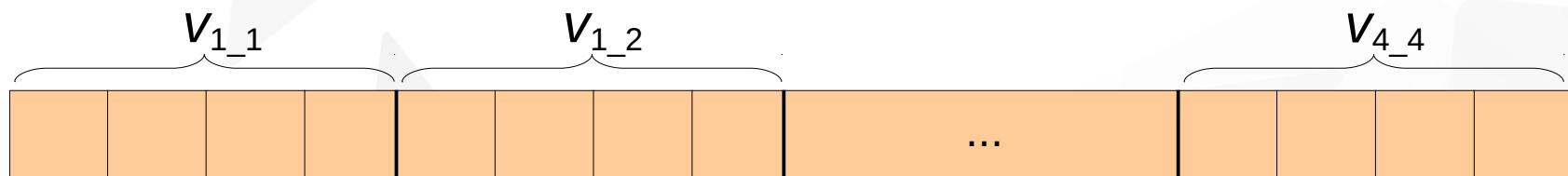
- El vector v representa el patrón de intensidades de la subregión:
 - Subregión homogénea: todos los elementos de v presentan valores bajos.
 - Frecuencias en la dirección horizontal: todos los elementos tienen valores bajos menos $\sum|d_x|$.
 - Intensidad creciente en la dirección horizontal: valores altos en $\sum d_x$ y $\sum|d_x|$.



Puntos característicos: descriptores de región

Descriptores Surf

- El descriptor es un vector formado por los distintos vectores v obtenidos por cada subregión.
- 4 componentes por subregión.
- 4x4 subregiones.
- 4x4x4 componentes para el descriptor.
- Descriptor = vector de 64 elementos.



Puntos característicos: descriptores de región

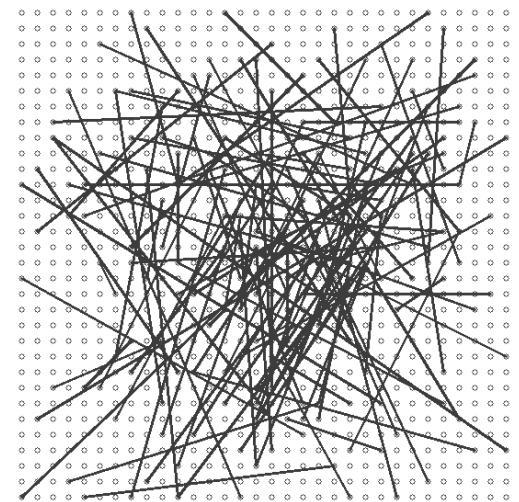
Descriptores ORB

- ▼ Un descriptor ORB (*rBRIEF*) es una versión orientada del descriptor *BRIEF*: **Binary Robust Independent Elementary Features**.
- ▼ El descriptor BRIEF se construye como una cadena binaria, donde cada bit indica el resultado de aplicar un determinado test sobre una pareja de puntos de la región.
- ▼ No es invariante a cambios de rotación.
- ▼ El descriptor rBRIEF proporciona invarianza a rotaciones aplicando la orientación de la región a los puntos antes de realizar cada test.

Puntos característicos: descriptores de región

Descriptores ORB

- ▼ Formación del descriptor BRIEF :
 - ▼ Suavizar la imagen: I_s
 - ▼ Por cada punto característico
 - ▼ Seleccionar 256 pares de puntos $p=(p_1, p_2)$ (a partir de un determinado patrón) dentro de una región cuadrada alrededor del punto.
 - ▼ Por cada par:
 - ▼ Si $I_s(p_1) < I_s(p_2)$, poner el bit correspondiente del descriptor a 1.
 - ▼ Sino, poner el bit a 0.

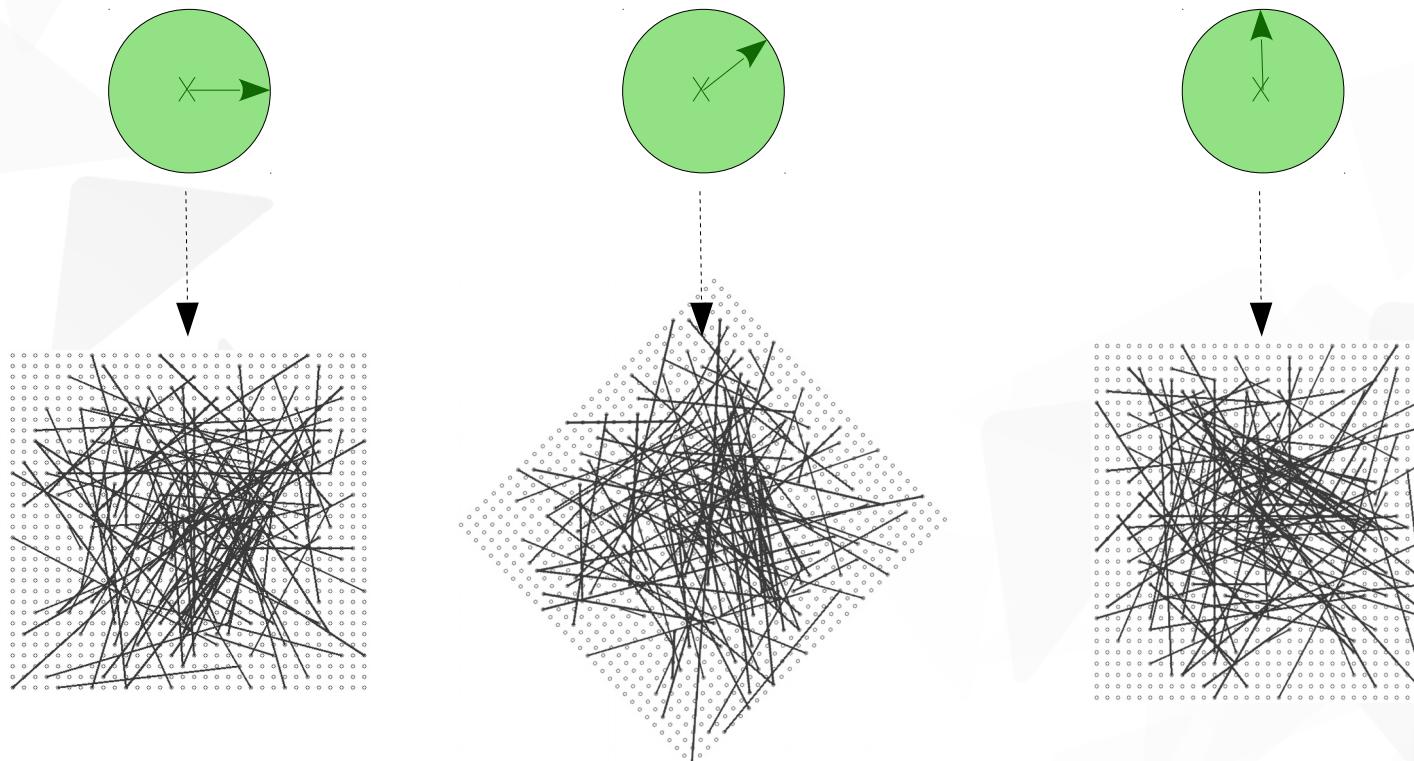


Patrón de pares de puntos

Puntos característicos: descriptores de región

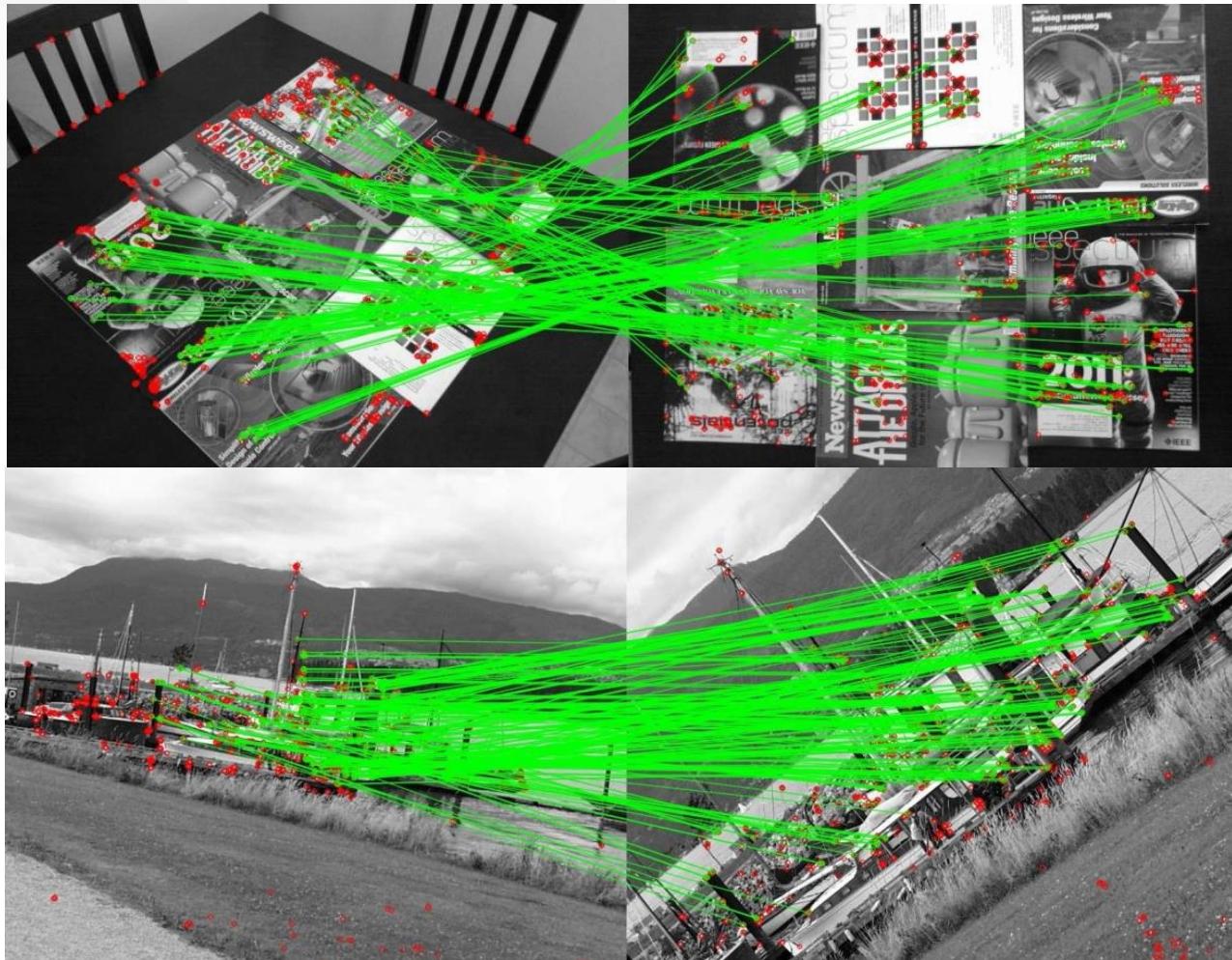
Descriptores ORB

- Para formar el descriptor rBRIEF, el conjunto de posiciones sobre las que se realizan los tests es rotado en base a la orientación de la región.



Puntos característicos: búsqueda de correspondencias

- ▼ **Objetivo:** emparejar regiones de dos imágenes con descriptores similares.



Puntos característicos: búsqueda de correspondencias

- ▼ Comparación de regiones: distancia entre descriptores. A menor distancia, mayor similitud.
- ▼ Distancia euclídea:

$$dist(d_1, d_2) = \sqrt{\sum_{i=0}^{N-1} (d_{1i} - d_{2i})^2}$$

- ▼ Distancia Manhattan:

$$dist(d_1, d_2) = \sum_{i=0}^{N-1} |d_{1i} - d_{2i}|$$

- ▼ Distancia de Hamming:

$$dist(d_1, d_2) = \text{número de elementos tales que } d_{1i} \neq d_{2i}$$

Puntos característicos: búsqueda de correspondencias

Generación de correspondencias

- ▼ Conjunto de regiones de la imagen prototipo: $R_p = \{(r_{pi}, d_{pi}), 0 \leq i < N_p\}$
- ▼ Conjunto de regiones de la imagen de búsqueda: $R_b = \{(r_{bj}, d_{bj}), 0 \leq j < N_b\}$
- ▼ Por cada región r_{pi} , buscar la región r_{bj} que minimiza $dist(d_{pi}, d_{bj})$.
- ▼ Aceptar la correspondencia $r_{pi} \rightarrow r_{bj}$ sólo si $dist(d_{pi}, d_{bj}) < maxDist$.
- ▼ Requisito adicional: dadas r_{bj} y r_{bk} , las dos regiones de mayor similitud con r_{pi} , aceptar la correspondencia $r_{pi} \rightarrow r_{bj}$ sólo si hay poca confusión:

$$\frac{dist(d_{pi}, d_{bj})}{dist(d_{pi}, d_{bk})} < 0.8$$

Detección de líneas y curvas

- ▶ Descripción del problema
- ▶ La transformada de Hough para líneas
- ▶ Generalización de la transformada de Hough

Descripción del problema

- ▼ A partir de una imagen de bordes, encontrar todas las instancias de una curva (líneas o elipses) o partes de ella (segmentos o arcos) presentes en la imagen.
- ▼ La detección de líneas y curvas conlleva resolver dos subproblemas:
 - ▼ *Agrupamiento*: determinar qué puntos de la imagen componen cada instancia de una curva.
 - ▼ *Ajuste del modelo*: por cada grupo de puntos obtenidos en el paso anterior, fijar los parámetros de la curva que mejor se ajusta al conjunto de puntos.

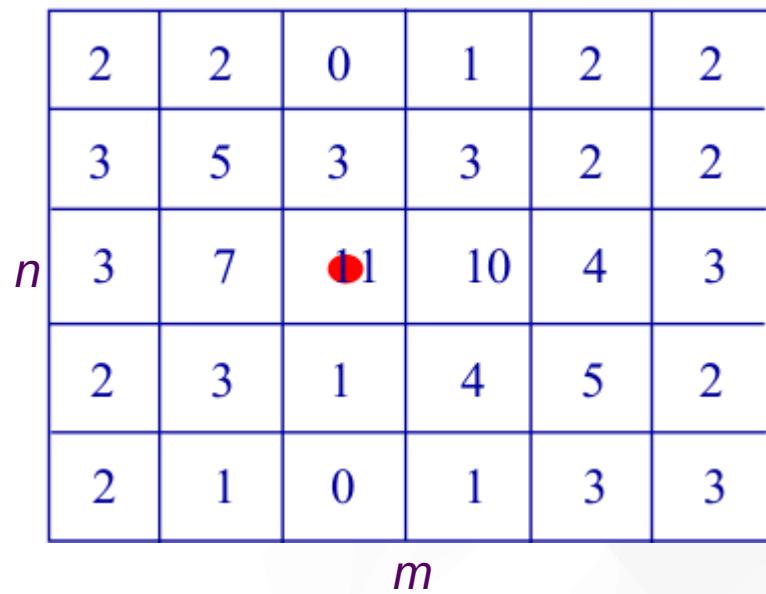
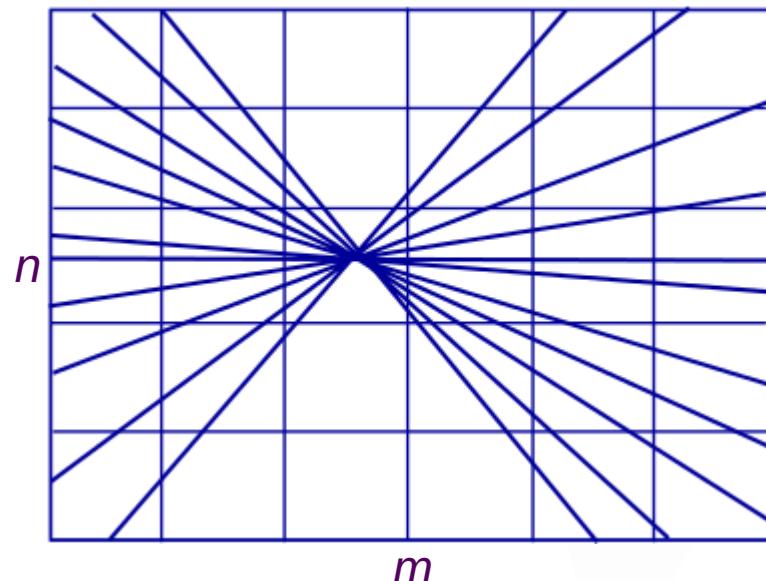
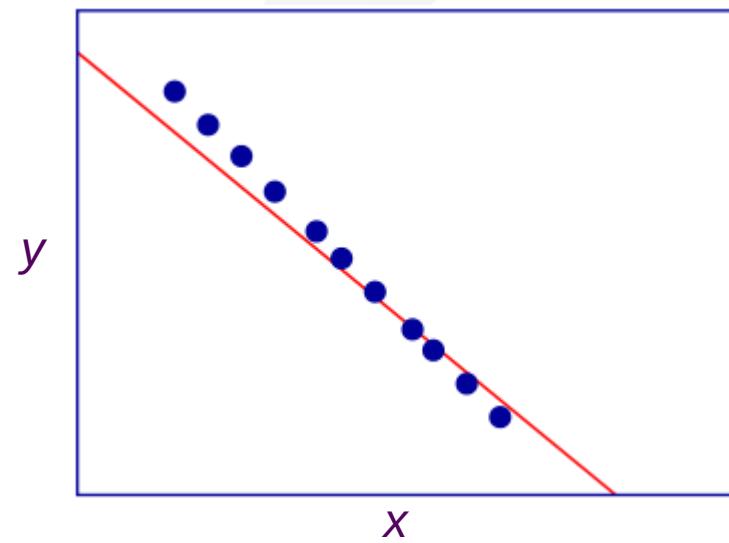
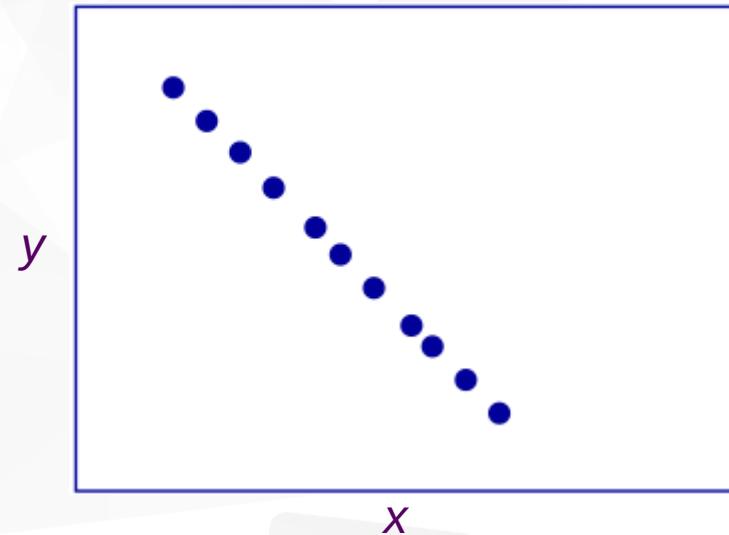
La transformada de Hough para líneas

- ▼ **Idea principal:** transformar el problema de detección de patrones (de línea) en un problema de detección de picos en el espacio de parámetros.
- ▼ El espacio de parámetros representa las posibles líneas que pueden estar presentes en la imagen.
- ▼ Cada punto de borde *vota* por todas las líneas a las que pertenece.
- ▼ La detección de líneas consiste en determinar los máximos locales del espacio de parámetros (*líneas más votadas*)

La transformada de Hough para líneas

- ▼ Una línea se representa como $y = mx + n$. Cada línea de la imagen se corresponde con un punto (m, n) en el espacio de parámetros.
- ▼ Cada punto de imagen se corresponde con una línea en el espacio de parámetros.
- ▼ Fijando (x,y) , (m,n) representan puntos que pertenecen a la línea $n = y - mx$. Es decir, todas las líneas que cruzan (x,y) en la imagen forman una línea en el espacio de parámetros.
- ▼ Los puntos pertenecientes a una misma línea en la imagen se corresponden con líneas que se cortan en el mismo punto del espacio de parámetros.

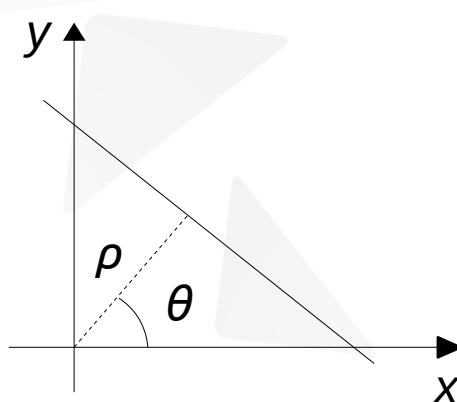
La transformada de Hough para líneas



La transformada de Hough para líneas

▼ Problemas:

- ▼ m es infinito en las líneas verticales (no se puede representar en un espacio discreto).
- ▼ Las líneas no se distribuyen uniformemente en el espacio de parámetros
- ▼ Solución – utilizar la representación polar de la línea



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

ρ : distancia del origen a la línea

θ : orientación de la línea

- ▼ En la representación polar, cada punto de imagen se corresponde con una sinusoida.

La transformada de Hough para líneas

▼ Algoritmo (*Incialización*)

- ▼ La entrada del algoritmo es una imagen binaria E de tamaño $M \times N$ resultante de aplicar un método de detección de bordes. Los píxels pertenecientes a un borde están marcados con un valor distinto de 0 y el resto tiene valor 0.
- ▼ El espacio de parámetros está definido por ρ y θ , donde $\rho \in [0, (N^2 + M^2)^{1/2}]$ y $\theta \in [0, \pi]$.
- ▼ Discretizar el espacio de parámetros considerando un tamaño de celda de $\delta\rho$ y $\delta\theta$.
- ▼ A nivel computacional el espacio de parámetros se representará mediante un array A compuesto por $L \times K$ celdas, siendo $L = \pi/\delta\theta$ y $K = (N^2 + M^2)^{1/2}/\delta\rho$.

La transformada de Hough para líneas

▼ Algoritmo (*Votación*)

- ▼ Sea A ($L \times K$) un array de contadores inicializados a 0.
- ▼ Por cada píxel $E(x, y) = 1$:
 - ▼ Para $l = 0, \dots, L-1$:
 - ▼ Calcular el valor de θ asociado con l : $\theta = l * \delta\theta$
 - ▼ Calcular $\rho = x \cos(\theta) + y \sin(\theta)$
 - ▼ Calcular el índice del array k asociado con ρ : $k = \rho / \delta\rho$
 - ▼ Incrementar $A(l, k)$ en 1.

La transformada de Hough para líneas

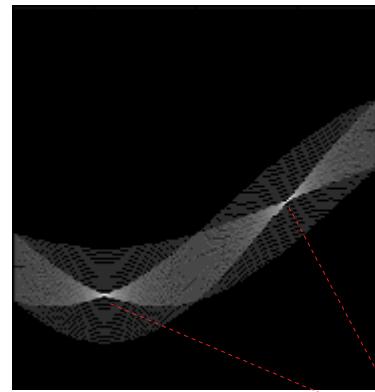
▼ Algoritmo (*Detección*)

- ▼ Detectar los máximos locales (l_p, k_p) de A , tales que $A(l_p, k_p) \geq \tau$, siendo τ el menor número de votos para considerar una línea.
- ▼ El resultado es un conjunto de líneas descritas por (ρ_p, θ_p) , siendo $\rho_p = k^* \delta \rho_p$ y $\theta_p = l^* \delta \theta_p$.

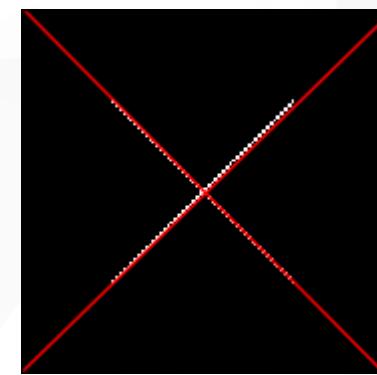
Imagen de bordes



Espacio de parámetros



Líneas detectadas



Generalización de la transformada de Hough

- ▶ La transformada de Hough puede generalizarse fácilmente para detectar curvas.
- ▶ Curvas: $y = f(x, a)$, siendo $a = [a_1, \dots, a_p]^T$ el vector de parámetros.
- ▶ El algoritmo es básicamente el mismo que para la detección de líneas suponiendo un espacio de parámetros formado por a_1, \dots, a_p .
- ▶ Limitación: el tiempo necesario para ejecutar el algoritmo crece exponencialmente con el número de parámetros.
- ▶ Alternativa al algoritmo básico: utilizar un espacio de parámetros de resolución variable.

Generalización de la transformada de Hough

- ▼ Algoritmo para la detección de curvas:
 - ▼ Sea $f(x,y,a) = 0$ la representación paramétrica de una curva.
 - ▼ Discretizar el espacio de parámetros a_1, \dots, a_p con intervalos constantes dando lugar a resoluciones de $\delta a_1, \dots, \delta a_p$ asociadas con los distintos parámetros.
 - ▼ Sea $A(k_1, \dots, k_p)$ un array de contadores inicializados a 0.
 - ▼ Por cada píxel (x, y) de la imagen de bordes E , tal que $E(x,y)=1$, incrementar todos los contadores de las curvas definidas por $f(x, y, a) = 0$ en A .
 - ▼ Localizar los máximos locales k_m tales que $A(k_m) \geq \tau$, para un cierto umbral τ .
 - ▼ El resultado es un conjunto de curvas descritas por $a_m = k_m * \delta a_m$.