

# FEATURE DETECTION SIFT

Andrés Godoy – Visión Computacional

# Objetivo:

- Extraer características distintivas invariantes. Correctamente emparejadas contra una gran base de datos de características extraídas de múltiples imágenes.
- Invariancia a la escala y rotación de la imagen
- Robustez frente a:
  - Distorsión afín,
  - Cambios en el punto de vista 3D,
  - Adición de ruido,
  - Cambios en la iluminación.

# SIFT- Scale Invariant Feature Transform

SIFT (propuesto por David Lowe en 1999 y perfeccionado en 2004) es uno de los métodos clásicos y más influyentes para llevar a cabo esta tarea. Provee:

1. Una forma de **detectar** esos puntos clave o keypoints.
2. Una **descripción** (descriptor) de la vecindad de cada punto, que lo hace **invariante** a la escala, a la rotación y a cambios moderados de iluminación.
- La palabra clave es *Scale Invariant* (invariante a la escala). Esto quiere decir que, si tomamos la misma escena con distinta distancia focal, o hacemos un zoom in/zoom out, el algoritmo reconocerá los mismos puntos de interés.

# Pasos

- Creación de espacios de escala
- Detección de keypoints
- Localización de keypoints
- Orientación de Keypoints
- Creación del descriptor

# Axiomas Fundamentales

## 1. Linealidad:

La construcción del espacio de escala es una operación lineal sobre la imagen original.

## 2. Invariancia ante traslaciones (Shift invariance):

No importa si desplazas una imagen antes o después del suavizado Gaussiano. El resultado será el mismo.

## 3. Invariancia ante rotaciones (Rotation invariance):

Si rotas la imagen original y luego aplicas el suavizado Gaussiano, obtendrás la misma imagen que si primero suavizas y luego rotas.

## 4. Isotropía:

No hay direcciones privilegiadas. La difusión o suavizado es igual en todas las direcciones, lo que implica que la Gaussiana es simétrica.

## 5. Principio de Semigrupo (Composición de Escalas):

Suavizar dos veces con Gaussianas es equivalente a suavizar una sola vez con una Gaussiana cuya varianza es la suma de las desviaciones cuadráticas:

$$G(x, y, \sigma_1) * G(x, y, \sigma_2) = G(x, y, \sqrt{\sigma_1^2 + \sigma_2^2})$$

# Primero: Definición del Espacio de Escala

Sea  $I(x, y)$  una imagen original en escala de grises. El espacio de escala  $L(x, y, \sigma)$  se define como una familia de imágenes suavizadas usando una función Gaussiana de desviación estándar  $\sigma$ :

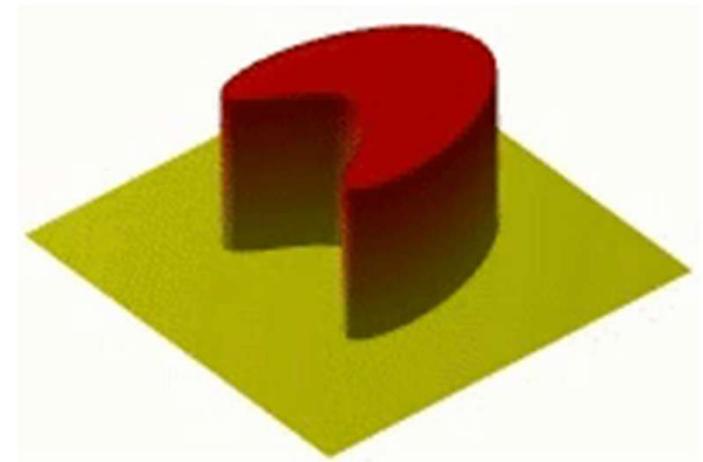
$$L(x, y, \sigma) = G(x, y, \sigma) * I(x, y)$$

donde  $*$  es el operador convolución y:

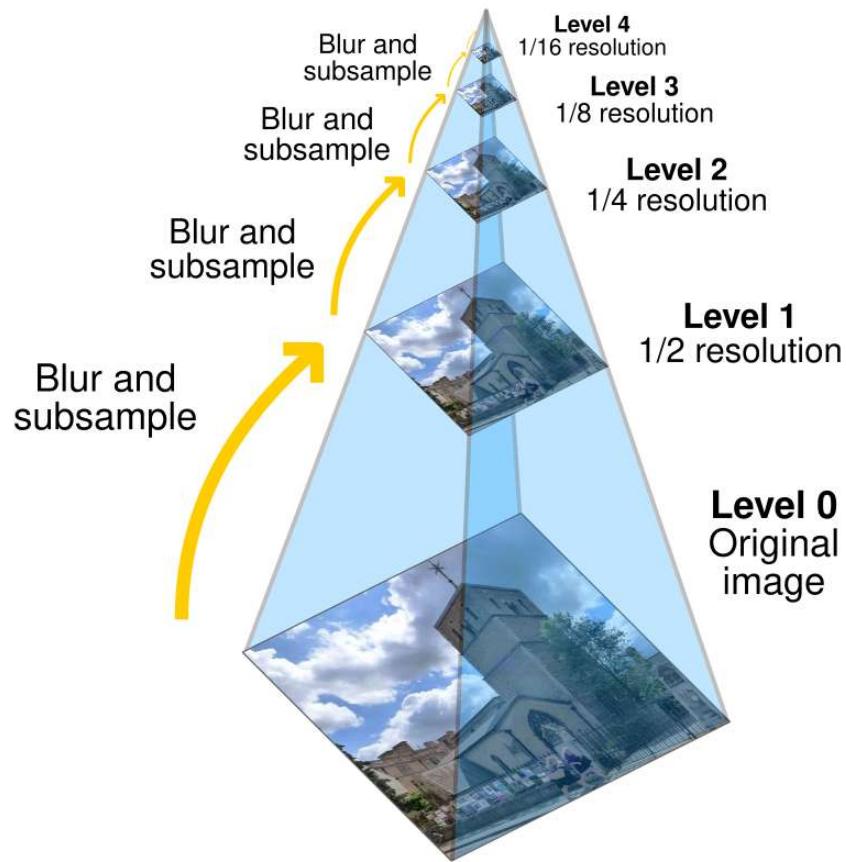
$$G(x, y, \sigma) = \frac{1}{2\pi\sigma^2} \exp\left(-\frac{x^2 + y^2}{2\sigma^2}\right)$$

# Relación con ecuación de calor

$$\frac{\partial L(x, y, \sigma)}{\partial \sigma} = \sigma \nabla^2 L(x, y, \sigma)$$



Irán permaneciendo patrones globales significativos.



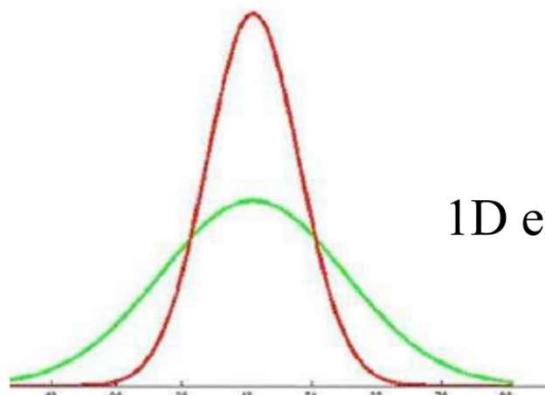
# Pirámide Gaussiana

Se suaviza un conjunto de imágenes (Octava) y luego, para la siguiente “octava superior” se reduce la resolución. Entonces es más de este estilo:

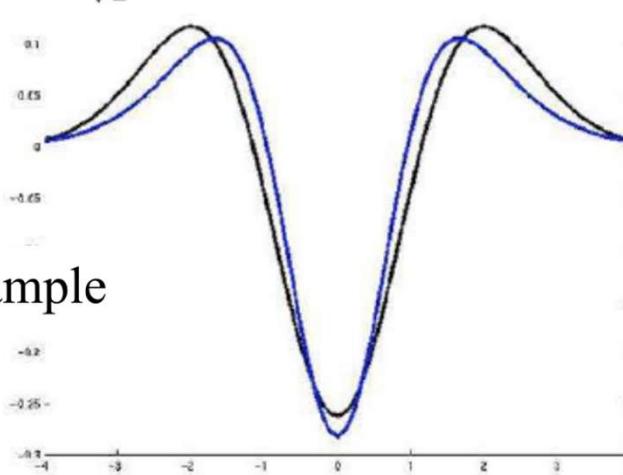


# Aproximamos LoG con DoG

$$\nabla^2 G_\sigma \approx G_{\sigma_1} - G_{\sigma_2}$$



Best approximation when:  
 $\sigma_1 = \frac{\sigma}{\sqrt{2}}$ ,  $\sigma_2 = \sqrt{2}\sigma$



## Segundo: Detección de máximos o mínimos (keypoint)

Para verificar si un punto es un máximo o mínimo local tridimensional:

1. Selecciona el punto central en la posición  $(x, y, \sigma)$ .
2. Compara su valor  $D(x, y, \sigma)$  con sus vecinos inmediatos en un **cubo de  $3 \times 3 \times 3$**  formado por:
  - Los **8 vecinos espaciales** en la misma escala (imagen actual),
  - Los **9 vecinos** en la imagen inmediatamente superior en escala (más difuminada, mayor  $\sigma$ ).
  - Los **9 vecinos** en la imagen de la escala inmediatamente inferior.

Esto da un total de  $8 + 9 + 9 = 26$  vecinos en total.

## Tercero: Keypoint Localization

- Tenemos un punto inicial detectado como máximo o mínimo en posición discreta.
- Nos movemos ligeramente en la dirección contraria al gradiente, ajustando según la curvatura local, para aterrizar justo en el "pico" exacto, que generalmente no coincide exactamente con una posición entera de píxel.

# 💡 ¿Por qué es importante tener una localización precisa (subpíxel)?

Cuando identificamos características en imágenes, estos puntos generalmente se usan para tareas posteriores que requieren una precisión muy alta, como:

1. Registro de imágenes (Image Alignment)
2. Reconstrucción 3D y Estimación de estructuras
3. Reconocimiento robusto de objetos
4. Calibración de cámaras y triangulación geométrica
5. Reconstrucción de panorámicas o "image stitching"

$$D(\mathbf{x}_0 + \Delta\mathbf{x}) \approx D(\mathbf{x}_0) + \underbrace{\nabla D(\mathbf{x}_0)^T \Delta\mathbf{x}}_{\text{Término lineal}} + \frac{1}{2} \underbrace{\Delta\mathbf{x}^T H(\mathbf{x}_0) \Delta\mathbf{x}}_{\text{Término cuadrático}}$$

Derivamos la aproximación respecto a  $\Delta\mathbf{x}$ :

$$\nabla D(\mathbf{x}_0 + \Delta\mathbf{x}) = \frac{\partial D(\mathbf{x}_0)}{\partial \mathbf{x}} + H\Delta\mathbf{x} = 0$$

$$\Delta\mathbf{x} = -H^{-1} \frac{\partial D(\mathbf{x}_0)}{\partial \mathbf{x}}$$

Una vez obtenida la corrección  $\Delta x$ , pueden pasar dos cosas:

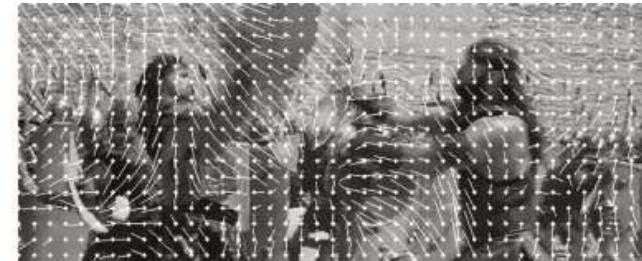
- **Ajuste pequeño:** Si el desplazamiento es pequeño, conservamos el keypoint y actualizamos su posición refinada a:
- Si el desplazamiento es demasiado grande (más de medio píxel de distancia), eso indica que el máximo o mínimo no es confiable, y el punto es descartado.
- También se pueden descartar puntos que tengan poca curvatura (que parezcan bordes), pues estos son menos estables (alta curvatura en una dirección, baja en otra).

# Cuarto: Orientación

Se com

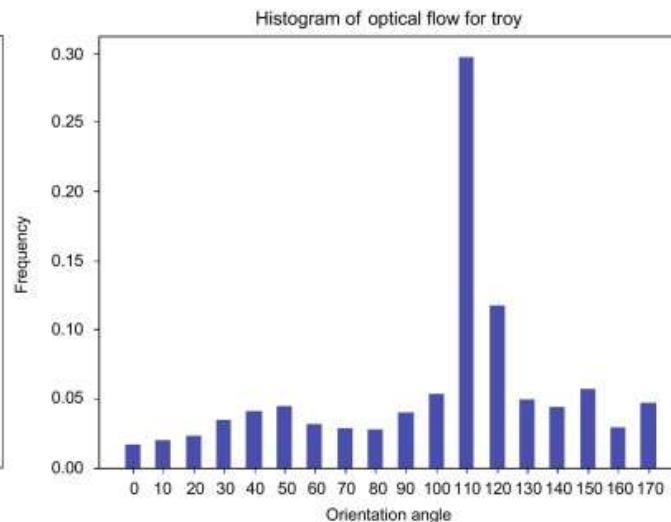
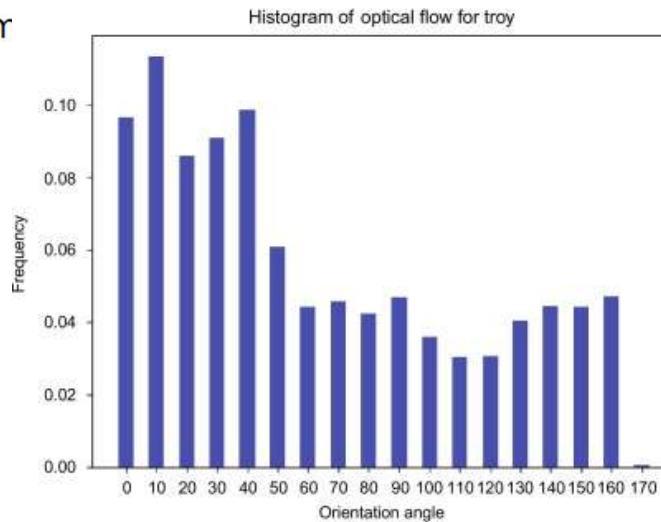
$m($

donde

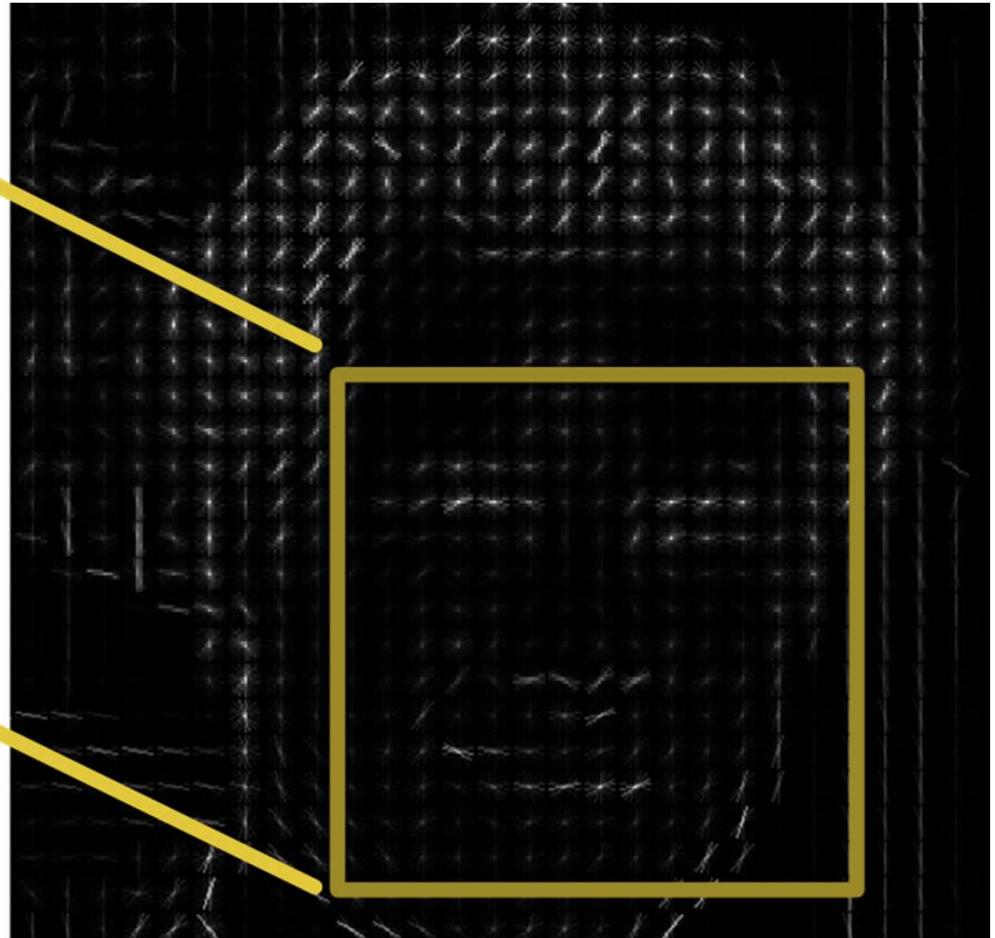
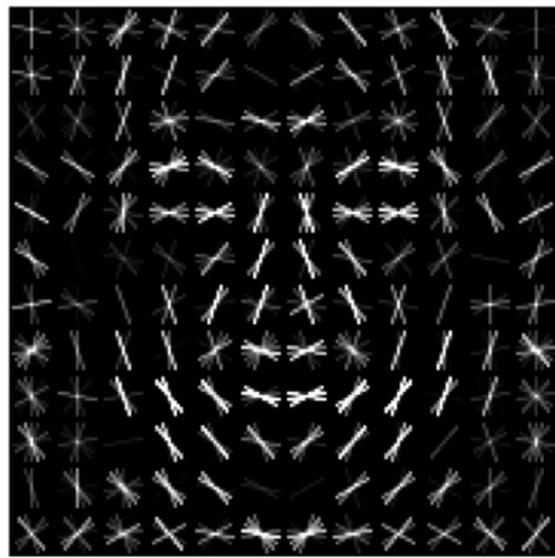


$, y)),$   
ente.

Se form



HOG face pattern generated  
from lots of face images



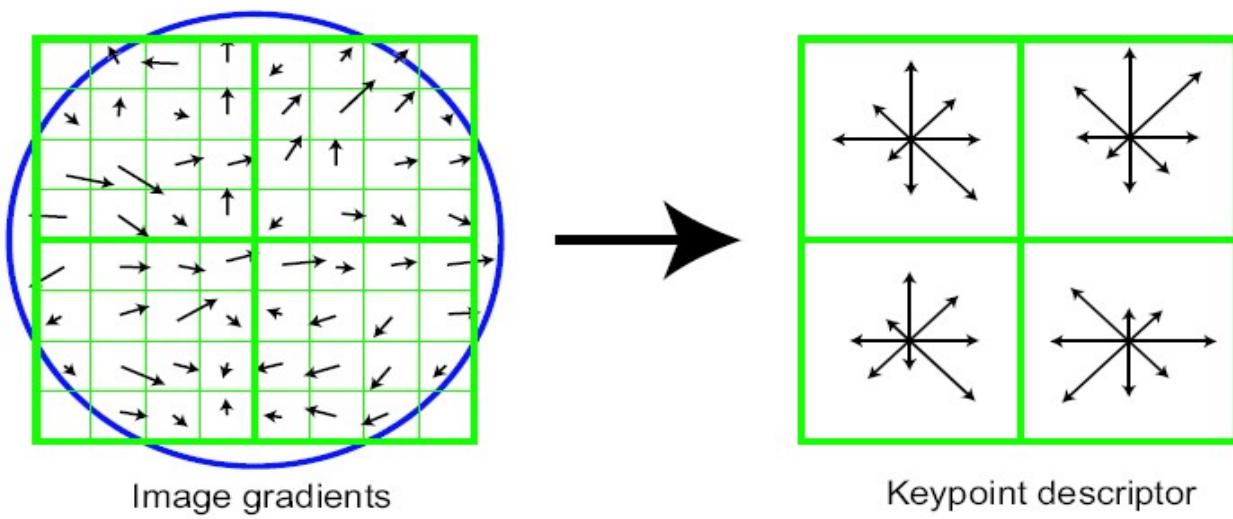
El pico más alto del histograma indica la orientación principal.

Si hay otros picos significativos (por encima de cierto umbral del pico principal), se pueden crear **múltiples keypoints** en la misma localización pero con orientaciones diferentes, de modo de captar estructuras con más de una dirección fuerte.

Así..... cada keypoint tiene ahora asociado  $(x,y,\sigma)$  y al menos una orientación  $\theta$ .

# Quinto: Creamos un descriptor

- Tomamos una región cuadrada alrededor del keypoint, generalmente de tamaño de **16x16 píxeles**
- Asegurando que todas las orientaciones estén medidas desde la dirección principal del keypoint ( $\theta = 0^\circ$ )
- La ventana rotada (16x16 píxeles) se divide en **subregiones de 4x4 (16 subregiones)**
- Se genera un histograma de orientaciones de los gradientes, típicamente con **8 direcciones**, es decir, "bins" cada uno de  $45^\circ$ .



Cada gradiente en la subregión vota en este histograma local según:

- La magnitud del gradiente determina el "peso" del voto.
  - La orientación del gradiente determina en cuál "bin" se asigna el voto.
- El resultado final es un vector de dimensión **128** (16 subregiones × 8 bins/subregión):

$$\mathbf{d}_{SIFT} = [h_{1,1}, h_{1,2}, \dots, h_{1,8}, h_{2,1}, \dots, h_{16,8}]$$