

# Descriptores de imágenes II

Visión por Computador, curso 2023-2024

---

Silvia Martin Suazo, [silvia.martin@u-tad.com](mailto:silvia.martin@u-tad.com)

10 de octubre de 2023

U-tad | Centro Universitario de Tecnología y Arte Digital

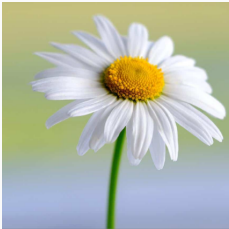


# Invarianzas

---

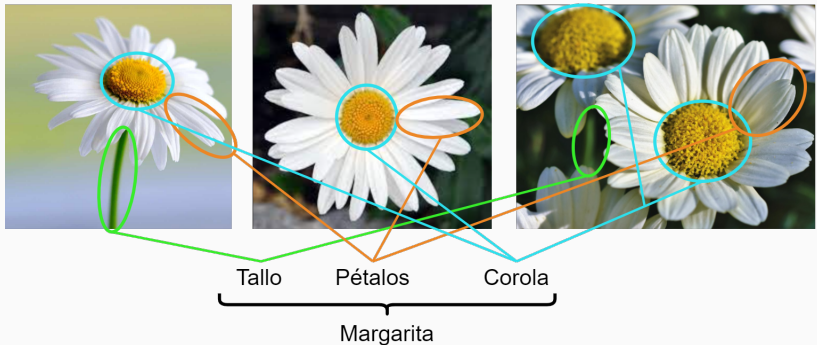
# La variabilidad de las imágenes

Uno de los mayores **problemas** a la hora de tratar con imágenes es que el **mismo objeto** puede aparecer de maneras muy **diversas** dependiendo de la imagen que se capture.



# El algoritmo “perfecto”

Un buen algoritmo que procese estas imágenes, debería ser capaz de **extraer** la información **relevante** de la imagen, a pesar de las **condiciones** en las que se capturó esta.



# Invarianzas

Estas **características** se conocen como **invarianzas** a distintos factores.

Una invarianza consiste en que un algoritmo **responde** de la **misma manera** ante distintas **variaciones** en distintos aspectos de una imagen.

Las invarianzas más importantes son:

- Invarianza a la translación
- Invarianza a la rotación
- Invarianza a la escala
- Invarianza a la iluminación

# Equivarianza

El término equivarianza es **muy similar** al de invarianza, sin embargo difieren en **detalles** que hacen que muchas veces se use **invarianza** de manera indistinta.

Un algoritmo **equivariante** debe tratar de la misma manera una imagen y su transformación, mientras que con la **invarianza** se busca obtener la misma **salida** para ambos casos.

Invarianza

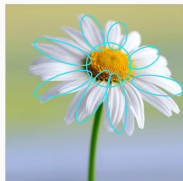


↓  
Margarita



↓  
Margarita

Equivarianza



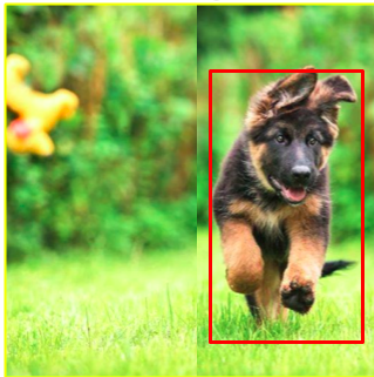
# Invarianza a la translación

Conocida como *translation invariance* o *shift invariance* depende de la posición del objeto dentro de la imagen.

train image

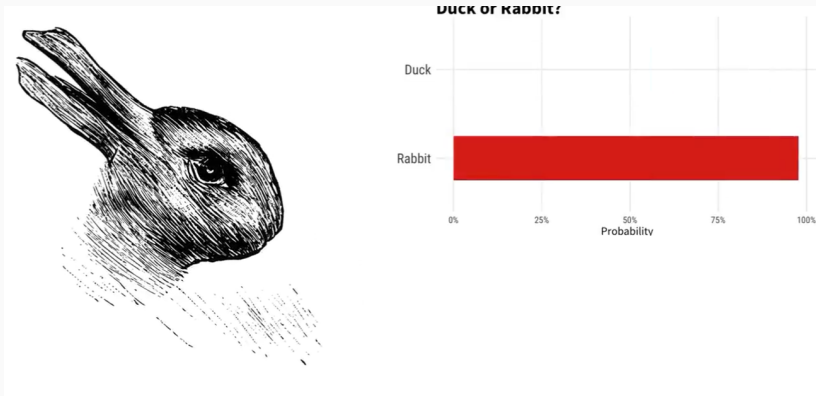


test image



# Invarianza a la rotación

La invarianza a la **rotación** depende del **ángulo** con el que se represente el objeto en la escena.

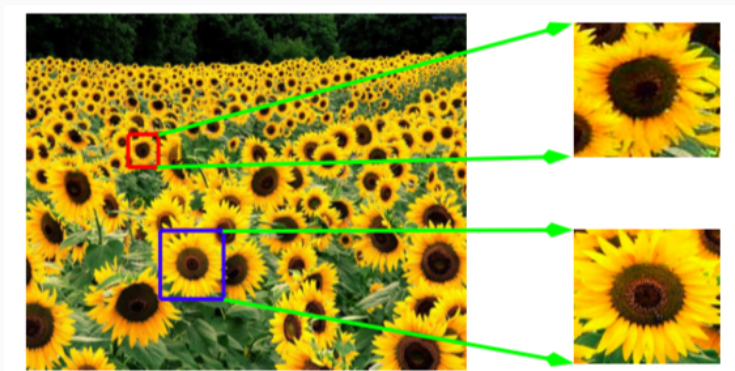


Vídeo twitter [1]



# Invarianza a la rotación

La invarianza a la **escala** depende del **tamaño** del objeto en la escena.



[2]

# Invarianza a la iluminación

La invarianza a la **iluminación** depende de las **condiciones fotográficas** con las que se ha capturado la imagen.



[3]

## Algoritmos de detección de puntos de interés II

---

# Notebook de ejemplos de descriptores

Los algoritmos que se explicarán a continuación pueden ser observados de manera práctica en el siguiente notebook.

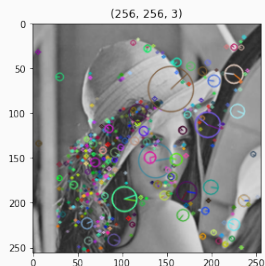
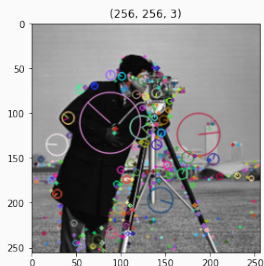
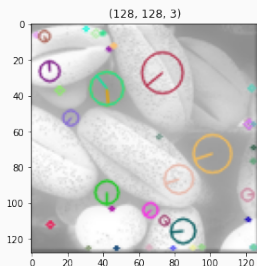


· 02.05-Descriptores.ipynb

# Algoritmo SIFT

El algoritmo **Scale-Invariant Feature Transform (SIFT)**[4] presenta otra aproximación para la **descripción de imágenes** centrada en tratar imágenes de distintas **escalas** de manera igual.

De esta manera, se pretende que **dos imágenes** iguales, pero de **distinto tamaño**, actúen de la misma **forma**.



El algoritmo puede dividirse en **cuatro pasos** diferenciados:

1. **Detección de puntos de interés:**

Para seleccionar una primera lista de **candidatos** para describir a la imagen original se aplica un algoritmo como la **Laplaciana de la Gaussiana** o la **diferencia Gaussiana**.

Esta identifica los **potenciales puntos de interés** a distintas escalas.

# Algoritmo SIFT

El algoritmo puede dividirse en **cuatro pasos** diferenciados:

## 2. **Localización de los puntos de interés:**

Durante este paso se realiza un **refinado** de los puntos detectados anteriormente.

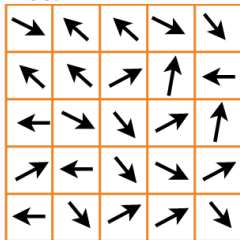
Para ello se utiliza el **determinante del Hessiano** de los puntos detectados anteriormente. Por las propiedades del Hessiano serán **eliminados** los puntos que sean **inestables**.



El algoritmo puede dividirse en **cuatro pasos** diferenciados:

### 3. **Orientación a los puntos de interés:**

Se crea un histograma de orientación con **36 barras** (debido a los  $360^\circ$  de giro). Se define un vecindario de puntos para cada punto de interés, para cada vecino se calcula su orientación. Por último, se crea una única orientación juntando las direcciones de los vecinos.



*\* con esto se consigue una invarianza a la rotación.*



El algoritmo puede dividirse en **cuatro pasos** diferenciados:

#### 4. **Generación del descriptor:**

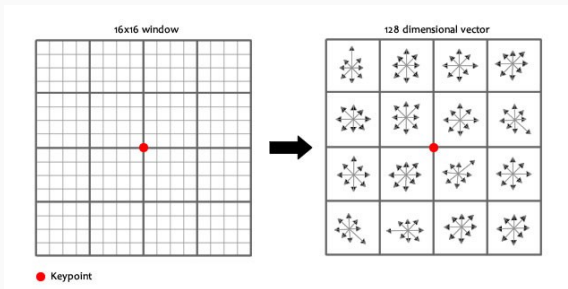
Para generar el **descriptor final** de la imagen se define un vecindario de **16x16 píxeles** alrededor de cada punto de interés. Esta región se divide en **16 bloques de 4x4**, para cada bloque se genera un histograma de orientación de 8 barras. Estas **128** barras generan un **vector** que describe cada **punto de interés**.

*\* con esto se consigue generar un descriptor único para cada punto de interés.*

# Algoritmo SIFT

El algoritmo puede dividirse en **cuatro pasos** diferenciados:

## 4. Generación del descriptor:



*\* con esto se consigue generar un descriptor único para cada punto de interés.*

5. *(Opcional)* **Correspondencia de puntos de interés:** Se utilizan técnicas como la distancia euclidiana para encontrar correspondencias de puntos clave entre dos imágenes.

El algoritmo de Speeded-Up Robust Features (SURF)[6] se centra en ser computacionalmente rápido respecto a sus predecesores.

La principal innovación de este algoritmo es el uso de filtros de caja, que permiten ser usado en tiempo real.

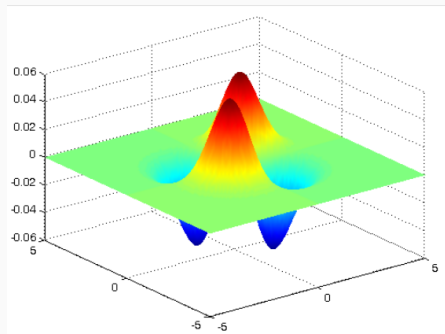
Este algoritmo funciona como una aproximación del Determinante del Hessiano.

El algoritmo puede dividirse en **cuatro pasos** diferenciados:

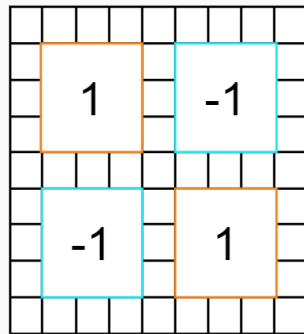
1. **Detección de puntos de interés:**

# Algoritmo SURF

Los **filtros de caja** presentados en este trabajo consisten en aproximaciones para las **derivadas parciales** de la función Gaussiana. Para ello se definen filtros de 9x9 píxeles. La **derivada parcial** respecto a  $xy$  se puede expresar como:



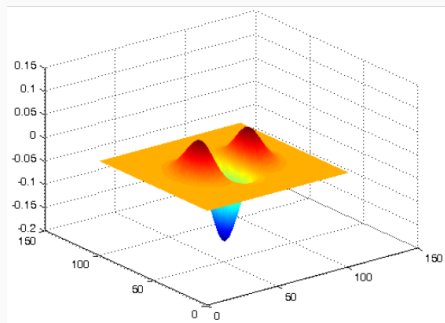
[7]



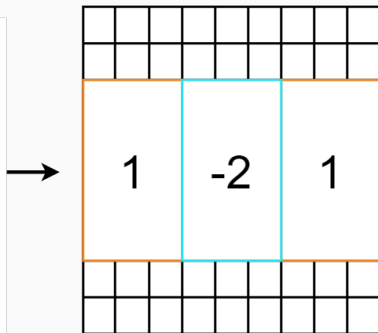
# Algoritmo SURF

Los **filtros de caja** presentados en este trabajo consisten en aproximaciones para las **derivadas parciales** de la función Gaussiana. Para ello se definen filtros de 9x9 píxeles.

La **derivada parcial** respecto a **xx** se puede expresar como:



[7]



De esta manera, el **determinante de la Hessiana** se puede calcular de manera aproximada a través de la siguiente ecuación:

$$\det(H_{\text{approx}}) = D_{xx}D_{yy} - (wD_{xy})^2 \quad (1)$$

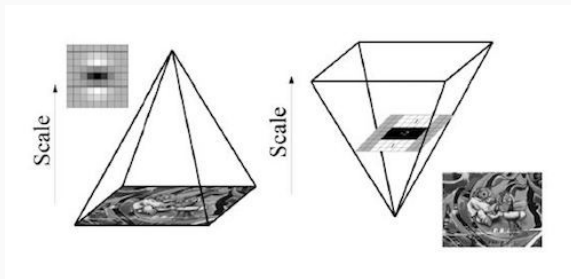
donde se **sugiere** que el valor de **w** sea de 0.9.



# Algoritmo SURF

Uno de los aspectos donde cobra especial **relevancia** la aproximación a las derivadas de la Gaussiana es a la hora de calcular puntos a distinta **escala**.

Para calcular los puntos de interés a **distintas escalas** basta con escalar el **filtro** de 9x9 píxeles a 15x15, 21x21, etc.



[8]

\* con esto se consigue invarianza en cuanto al tamaño.

El algoritmo puede dividirse en **cuatro pasos** diferenciados:

1. Detección de puntos de interés.
2. **Determinación de la orientación**: se asigna una orientación a cada punto identificado para el descriptor sea invariante a la rotación.
3. **Generación del descriptor**: se general filtros basandose en filtros Haar.
4. *(Opcional)* Coincidencia de puntos clave.

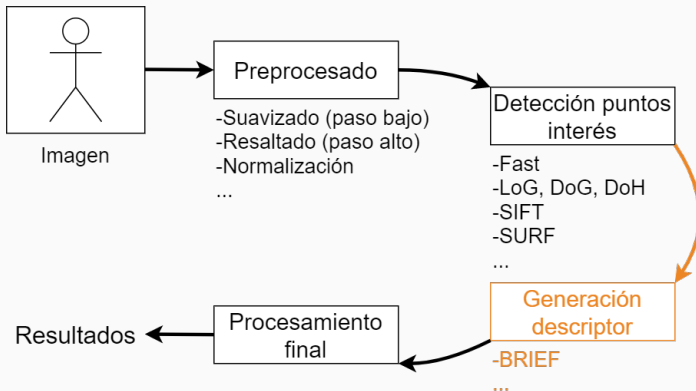
# Algoritmos de descripción de imágenes

---

# Descripción de imágenes en el procesamiento de imágenes

Una vez las estructuras **más relevantes** han sido extraídas de las imágenes, es necesario **organizar** dicha información para poder ser tratada adecuadamente.

Este paso **prepara** la información que se ha obtenido anteriormente para **ser procesada** y generar los **resultados finales**.



El algoritmo **Binary Robust Independent Elementary Features (BRIEF)** es un algoritmo para la **generación de descriptores de imágenes**.

Dentro de la metodología que se está siguiendo, a través de algoritmos como este, se consiguen establecer **relaciones** entre **imágenes similares**.

Los **puntos de interés** detectados anteriormente son **descritos** para su posterior relación.

# Algoritmo BRIEF

El algoritmo BRIEF basa la descripción de imágenes a través de **vectores binarios** que definen los **puntos de interés detectados previamente** en la imagen.

Estos tienen una **longitud** entre **128-512 bits** normalmente.

$$\begin{array}{l} \text{Vectores de} \\ \text{características} \\ \text{binarios} \end{array} \left\{ \begin{array}{l} V_1 = [01010011\dots] \\ V_2 = [11110000\dots] \\ V_3 = [00011011\dots] \\ \dots \\ V_n = [01010011\dots] \end{array} \right.$$

El algoritmo es **sensible** a la aparición de **ruido**, por lo tanto el primer paso que se toma es el **suavizado** de la imagen a través del uso de un **filtrado Gaussiano** de  $\sigma$  entre 1 y 3.

Posteriormente, cada vector se define de la siguiente manera:

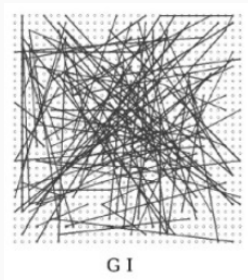
$$\tau(p; x, y) = \left\{ \begin{array}{ll} 1 & : p(x) < p(y) \\ 0 & : p(x) \geq p(y) \end{array} \right\} \quad (2)$$

donde  $p(x)$  es la intensidad de el punto de referencia  $x$ ,  $y$  es el punto con el que se realiza la comparación.

# Algoritmo BRIEF

La selección de **pares** que forman parte de cada **vector binario** se puede realizar de diversas maneras para un parche de tamaño  **$S \times S$** :

- **Uniforme (G I)**: Los píxeles  **$x$**  e  **$y$**  se escogen a través de una **distribución uniforme** de extensión  **$S/2$** .



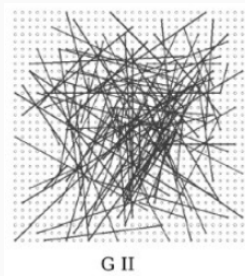
G I  
[9]



# Algoritmo BRIEF

La selección de **pares** que forman parte de cada **vector binario** se puede realizar de diversas maneras para un parche de tamaño  $S \times S$ :

- **Gaussiana (G II)**: Los píxeles  $x$  e  $y$  se escogen a través de una **distribución Gaussiana** de extensión  $0.04 * S^2$ .



**G II**

[9]

# Algoritmo BRIEF

La selección de **pares** que forman parte de cada **vector binario** se puede realizar de diversas maneras para un parche de tamaño  $S \times S$ :

- **Gaussiana (G III)**: El primer píxel  $x$  se escoge a través de una **distribución Gaussiana** de extensión  $0.04 * S^2$  y el segundo píxel  $y$  a través de una **distribución Gaussiana** de extensión  $0.01 * S^2$  centrada en el primer píxel  $x$ .

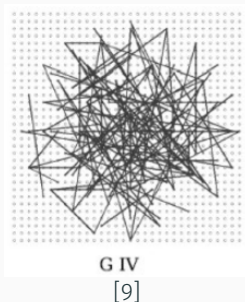


[9]

# Algoritmo BRIEF

La selección de **pares** que forman parte de cada **vector binario** se puede realizar de diversas maneras para un parche de tamaño **SxS**:

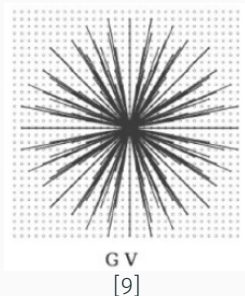
- **Coarse Polar Grid (G IV)**: Los píxeles **x** e **y** se escogen a través de coordenadas polares.



# Algoritmo BRIEF

La selección de **pares** que forman parte de cada **vector binario** se puede realizar de diversas maneras para un parche de tamaño **SxS**:

- **Coarse Polar Grid (G V)**: El primer píxel **x** siempre está centrado en (0, 0) y el segundo píxel **y** se escoge a través de una **coordenadas polares**.



- [1] Max Woolf @minimaxir (Twitter).  
**Rotation invariance video.**  
[Online; accessed August, 2022].
- [2] Minghao Ning (Towards Data Science).  
**Scale invariance image.**  
[Online; accessed August, 2022].
- [3] Wenyi Zhao and Rama Chellappa.  
***Face Processing: Advanced modeling and methods.***  
Elsevier, 2011.
- [4] David G Lowe.  
**Distinctive image features from scale-invariant keypoints.**  
*International journal of computer vision*, 60(2):91–110, 2004.

- [5] Geeks for Geeks.  
**Sift before and after keypoint localization image.**  
[Online; accessed August, 2022].
- [6] Herbert Bay, Andreas Ess, Tinne Tuytelaars, and Luc Van Gool.  
**Speeded-up robust features (surf).**  
*Computer vision and image understanding*, 110(3):346–359, 2008.
- [7] Technische Universität München.  
**Gaussian derivative images.**  
[Online; accessed August, 2022].
- [8] Deepanshu Tyagi (Medimum).  
**Surf pyramid image.**  
[Online; accessed August, 2022].

- [9] Deepanshu Tyagi (Medimum).  
**Brief distribution images.**  
[Online; accessed August, 2022].