



Recuperación de Información Multimedia

Codebooks

CC5213 – Recuperación de Información Multimedia

Departamento de Ciencias de la Computación

Universidad de Chile

Juan Manuel Barrios – <https://juan.cl/mir/> – 2020

Búsqueda de objetos

- ¿Cómo buscar las apariciones de ciertos objetos conocidos en uno o más videos?
 - Se tiene una o más imágenes de ejemplo para cada objeto buscado
 - Supuestos: Q imágenes de consulta, cada una con N_q descriptores locales. Video de M frames, cada frame con N_f descriptores locales
 - $Q \sim 10$, $N_q \sim 1000$, $M \sim 10000$, $N_f \sim 1000$



Búsqueda de objetos





Descriptores Locales

■ Opción 1:

- De cada frame se obtienen sus descriptores y se comparan con los descriptores de la consulta
- Se utiliza la coherencia espacial de los calces para decidir la ocurrencia del objeto
- Buscar el objeto requiere $Q N_q N_f M$ distancias más $Q M$ cálculos de coherencia espacial

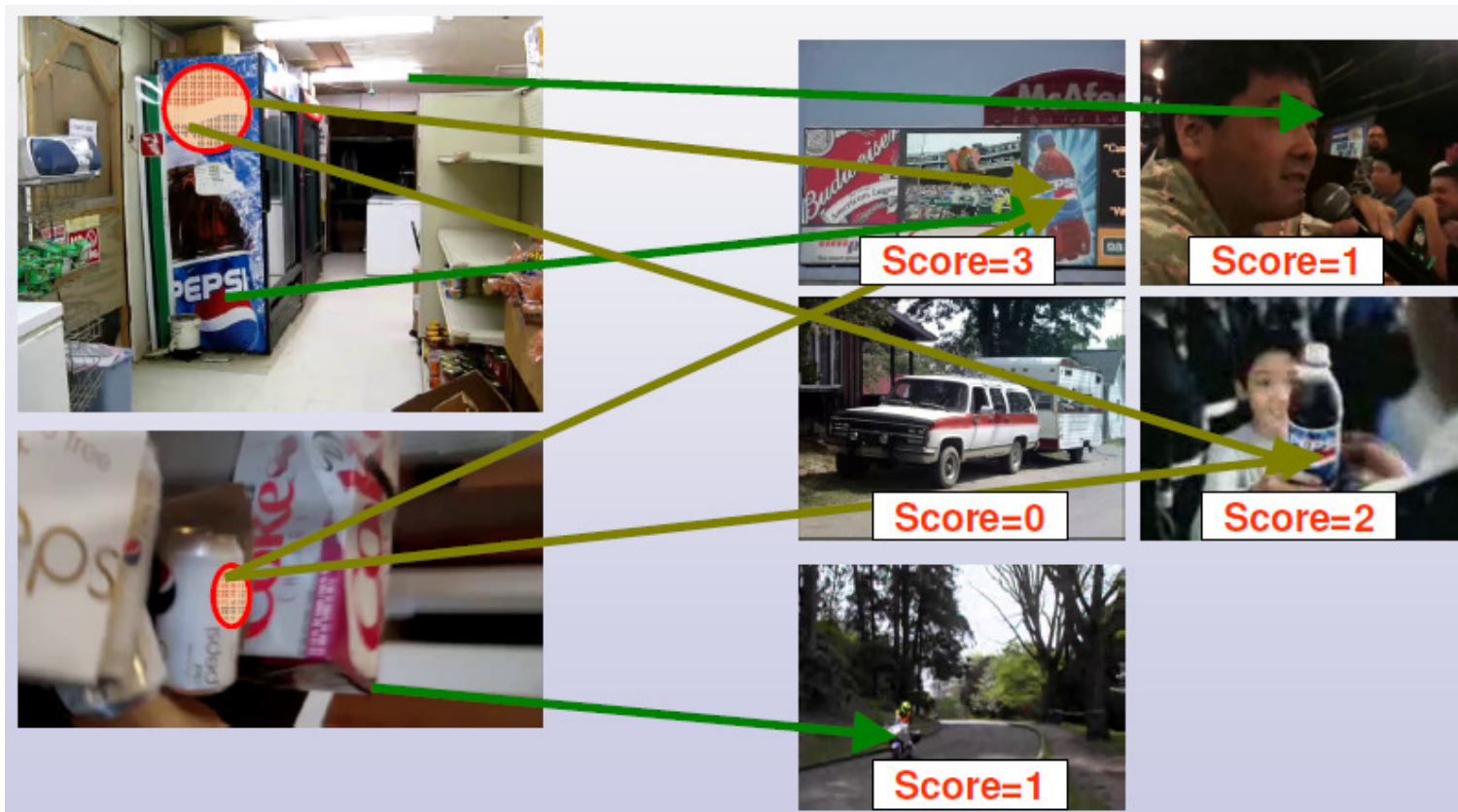


Búsquedas aproximadas

■ Opción 2:

- Calcular todos los descriptores locales en el video
- Construir un índice con los descriptores de todo el video
- Para cada descriptor de cada imagen de consulta realizar una búsqueda k-nn aproximada
- Cada vecino más cercano encontrado suma un voto al frame que lo contiene
 - Ponderar el voto por el rank y distancia
- Los frames con más votos deben contener el objeto
- Construir un índice con N_f M vectores y realizar Q N_q búsquedas k-nn aproximadas

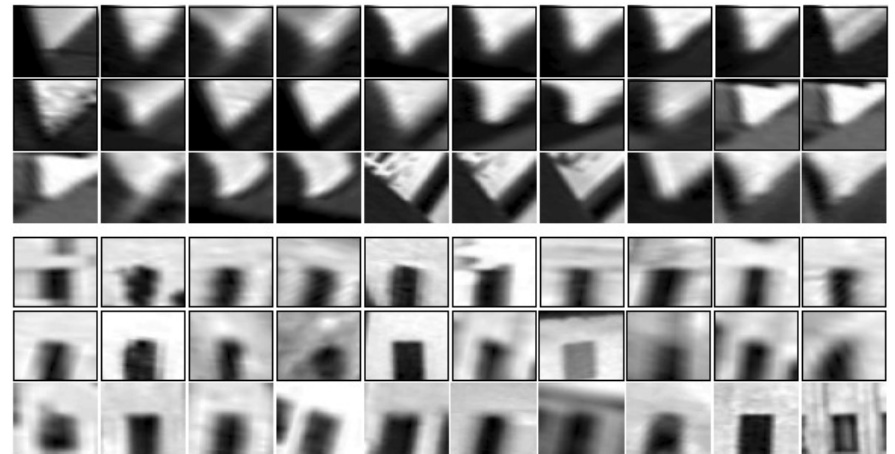
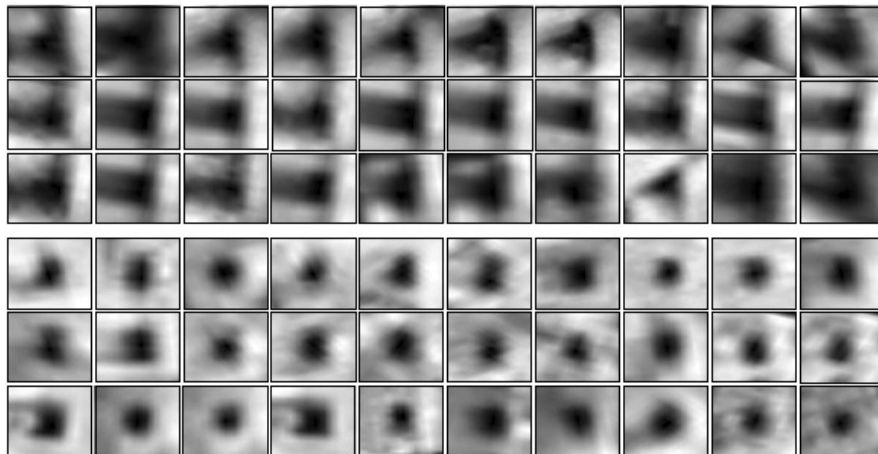
Búsquedas aproximadas



Palabras Visuales

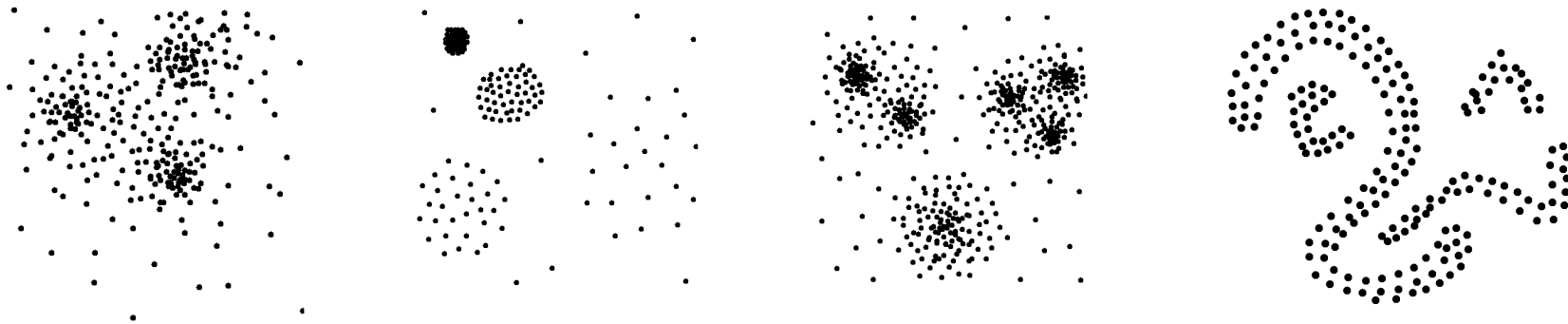
■ Opción 3:

- Si un mismo objeto aparece en varios frames del video, lo más probable es que una gran cantidad de descriptores locales sean muy parecidos entre sí
- Convertir el espacio continuo de descriptores a una cantidad discreta de descriptores (vocabulario visual)



Clustering

- Definir un conjunto finito de clusters (grupos, categorías o clases) de la colección de datos tales que:
 - Objetos en el *mismo* cluster deben ser lo más similares posible
 - Objetos en *diferentes* clusters deben ser lo más disímiles posible



- **Propiedades de los cluster:**
 - clusters pueden tener tamaños, formas y densidades diferentes
 - clusters pueden formar una jerarquía
 - clusters pueden traslaparse o ser disjuntos

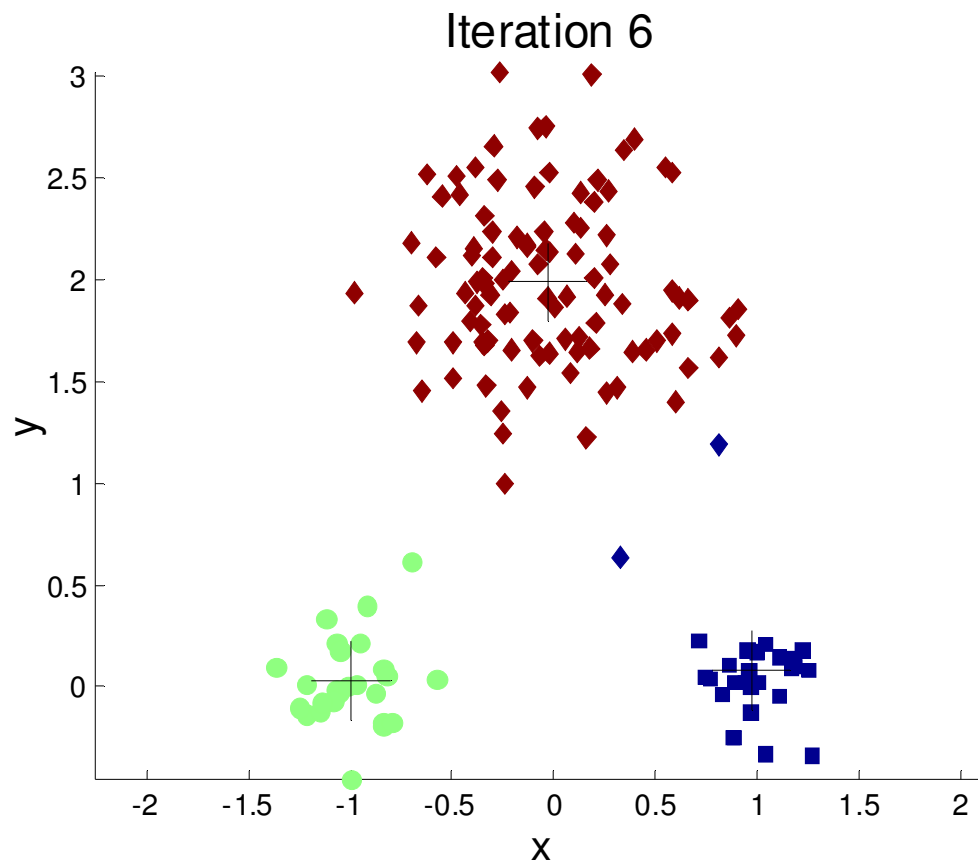


K-means

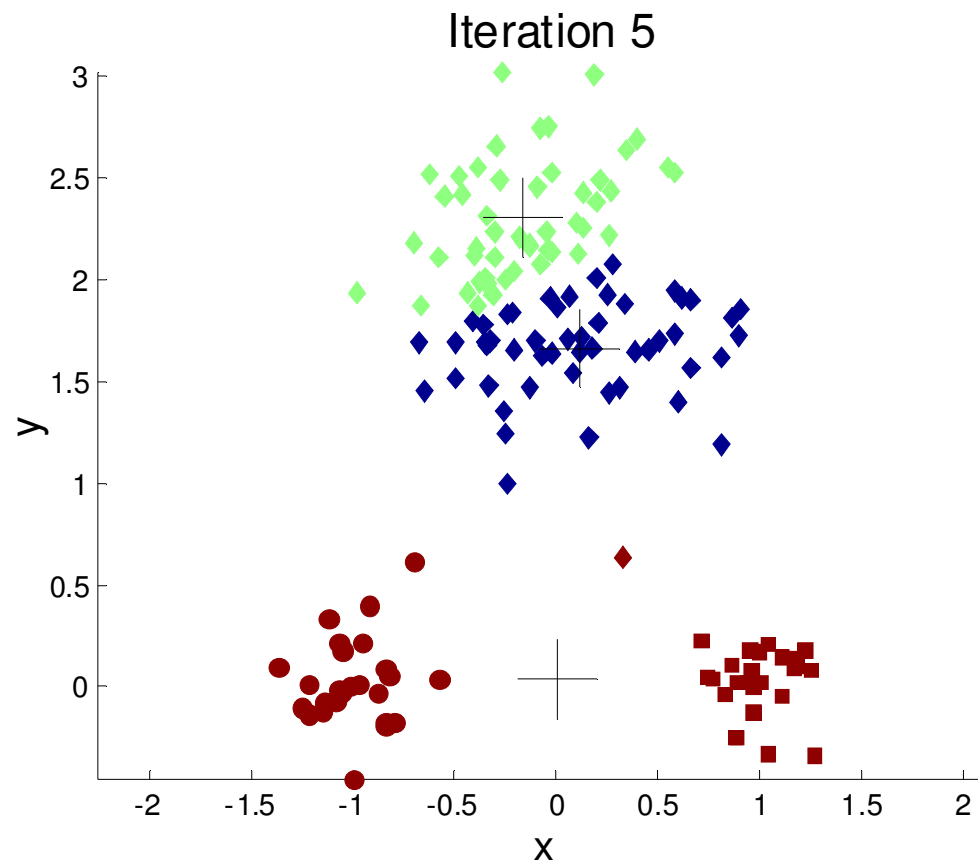
- Cada cluster se asocia con un centroide (punto central)
- Cada punto es asignado al cluster con el centroide más cercano
- El número de clusters K debe ser especificado

-
- 1: Select K points as the initial centroids.
 - 2: **repeat**
 - 3: Form K clusters by assigning all points to the closest centroid.
 - 4: Recompute the centroid of each cluster.
 - 5: **until** The centroids don't change
-

K-means Ejemplo 1



K-means Ejemplo 2





K-means

■ Ventajas

- Relativamente eficiente: $O(t \cdot k \cdot n)$, donde: **n** es el número de vectores, **t** número de iteraciones ($t \ll n$), **k** es número de centroides
- Fácil de implementar, paralelismo

■ Desventajas

- Muchas veces termina en un óptimo local
- Utilizable sólo cuando el promedio está definido (no se puede utilizar en espacios métricos generales)
- Enfocado en identificar clusters circulares
- Necesita especificar **k** (número de clusters) como parámetro
- No es robusto a ruido, outliers, densidades distintas

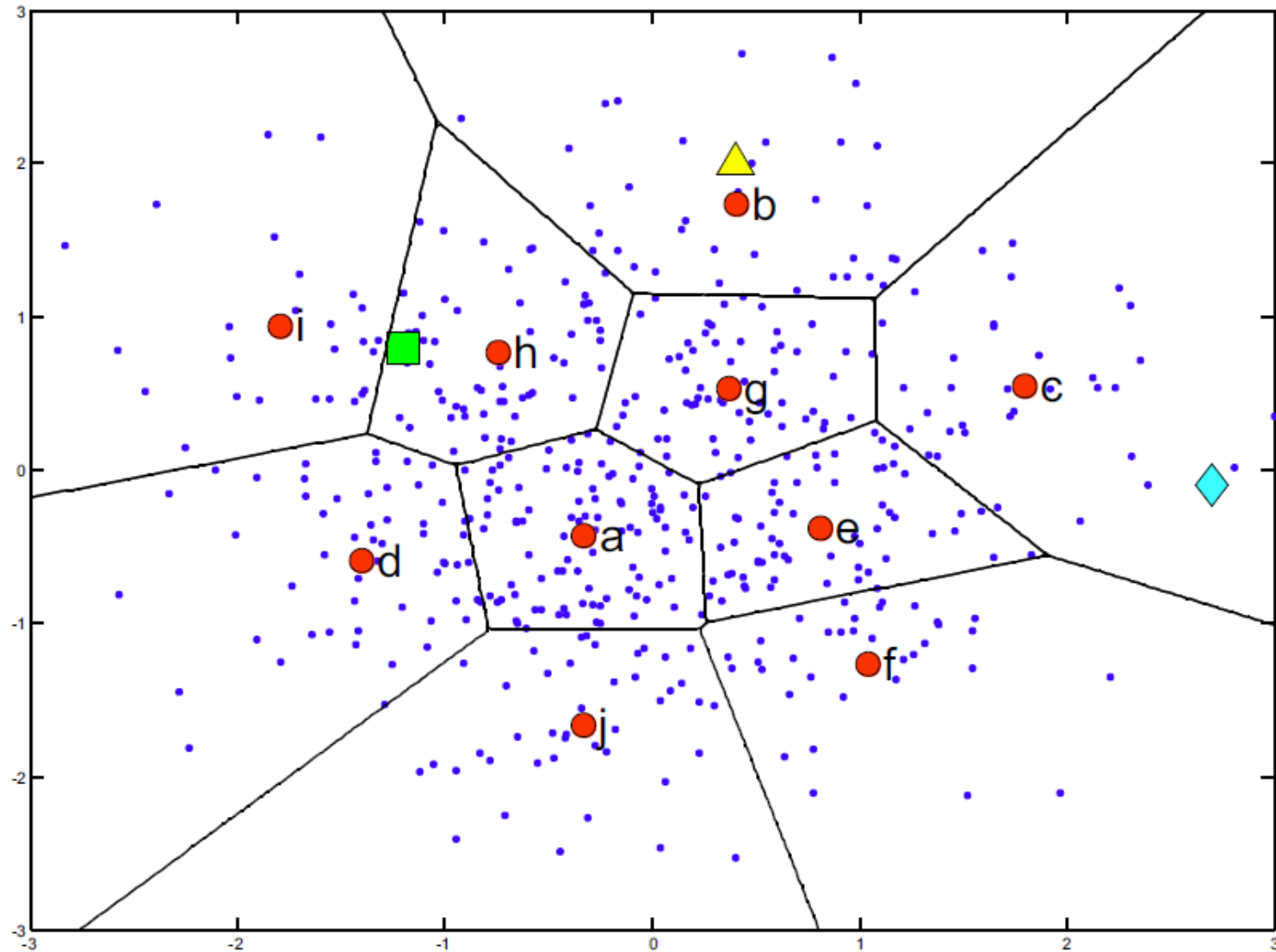



Cálculo de palabras visuales

- Dado un conjunto de descriptores locales calcular K clusters para formar un “**vocabulario visual**” o “**codebook**”
 - K-Means puede escalar a grandes cantidades de vectores.
- Cada centroide representa una “**palabra visual**”
- Se reemplaza cada descriptor local por la palabra visual más cercana.
- El tamaño del vocabulario incide en su representatividad (efectividad) y tiempo de clusterización y búsqueda (eficiencia)



Palabras visuales o Codebook





Bag-of-Visual-Words (BOVW)

■ BOVW:

- Descriptor global como agregación de descriptores locales
- El descriptor de la imagen es un resumen de las “palabras visuales” que contiene
- La búsqueda se basa en localizar los frames que comparten más “palabras visuales” con la consulta
- Se puede usar un **índice invertido** para localizar frames con más coincidencias con la imagen de consulta

Bag-of-Visual-Words (BOVW)

- Comparación de imagen de consulta con frames usando similitud coseno
- Usar tf-idf para favorecer palabras visuales discriminativas y evitar palabras visuales ruidosas
 - N =número de imágenes en la colección
 - n_i =número de imágenes conteniendo la palabra i
 - $freq_{ij}$ =cantidad de ocurrencias de la palabra i en la imagen j
 - n_d =número de palabras en la imagen d

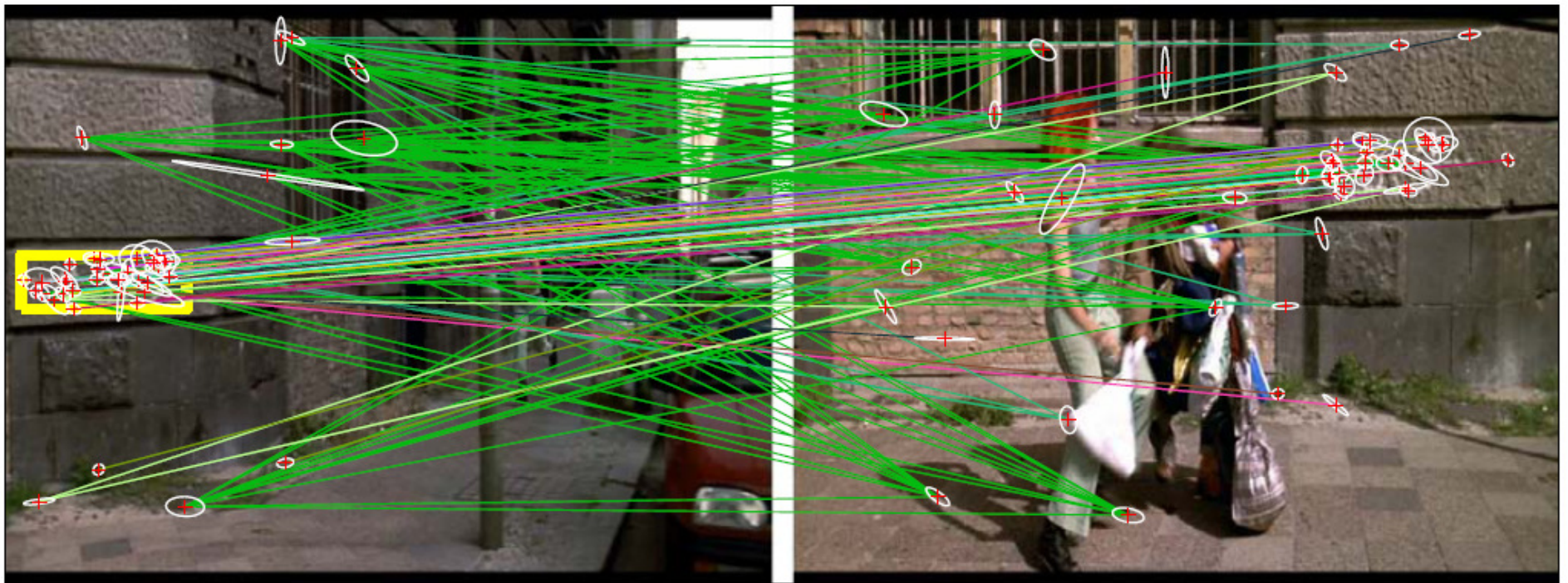
$$(t_1, \dots, t_i, \dots, t_k) \quad w_{ij} = \begin{cases} (1 + \log(freq_{ij})) \times \log(\frac{N}{n_i}) & \text{si } freq_{ij} > 0 \\ 0 & \text{si no} \end{cases}$$



Coherencia Espacial

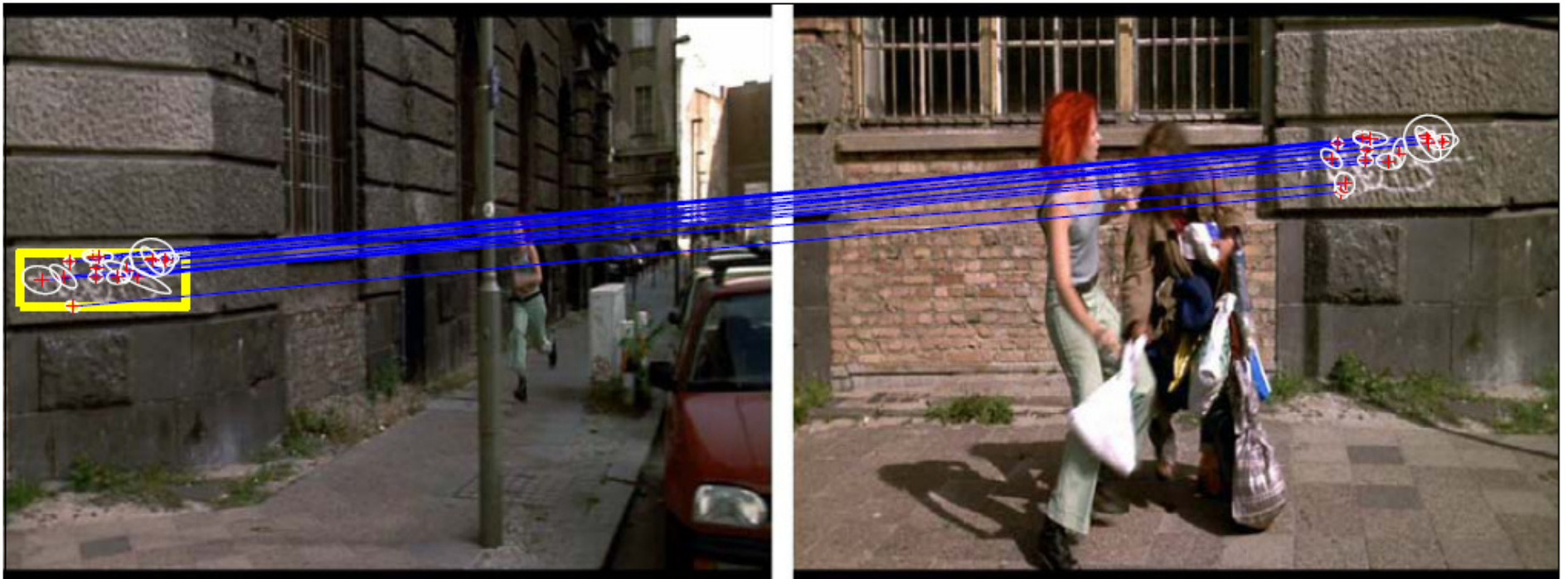
- Entre los frames candidatos (obtenidos por tf-idf) se deben preferir los que contienen palabras visuales en lugares parecidos a la consulta
- Opción 1: Realizar RANSAC entre frames y consulta
 - Los matches iniciales son las coincidencias de palabras visuales
 - Lento de ejecutar para muchos candidatos
- Opción 2: Usar ubicación relativa entre palabra visuales precalculado en el Índice Invertido
 - En cada entrada del Índice Invertido junto con la palabra visual guardar palabras visuales cercanas espacialmente en el frame
 - En la imagen de consulta, para cada palabra visual determinar además las palabras visuales cercanas espacialmente
 - Al buscar favorecer las coincidencias que además tienen alta intersección entre sus listas de palabras visuales cercanas

Coincidencia de palabras visuales



Sivic, Zisserman. "Video Google: A Text Retrieval Approach to Object Matching in Videos", 2003.

Coherencia espacial de palabras visuales



Sivic, Zisserman. "Video Google: A Text Retrieval Approach to Object Matching in Videos", 2003.

Demo

<http://www.robots.ox.ac.uk/~vgg/research/vgoogle/index.html>

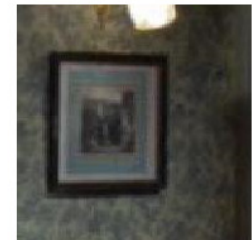


Matched image, frame 84475



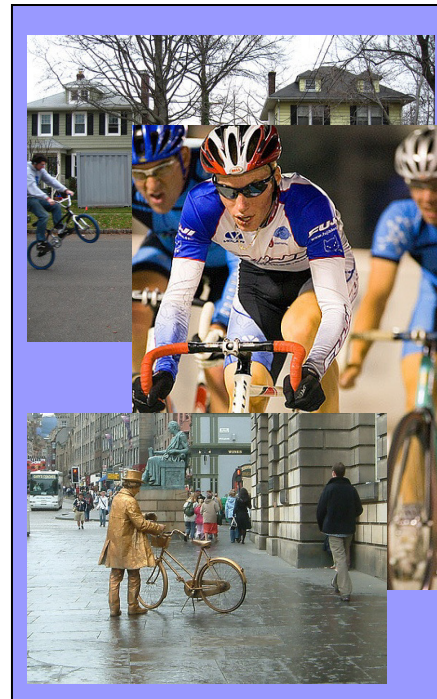
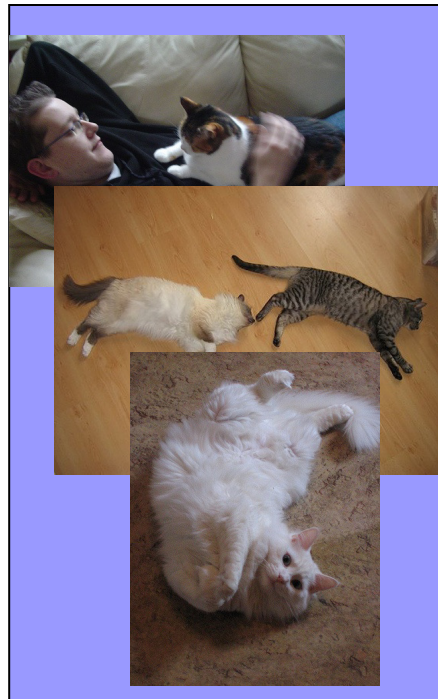
Matched region from frame 84475

Original selection from frame 130750



Clasificación de imágenes

- Dadas N clases (ej. aviones, gatos y bicicletas) con varias imágenes de entrenamiento, determinar la clase de una imagen de consulta
 - Ej: PASCAL VOC Challenge <http://host.robots.ox.ac.uk/pascal/VOC/>



??



Clasificación de imágenes

- Imágenes de una misma clase (i.e., que muestran un mismo tipo de objeto) debieran compartir zonas con formas parecidas
 - Por ejemplo, ruedas en la clase bicicleta, alas en aviones, orejas en gatos, etc.
 - Al calcular descriptores locales, las imágenes de una misma clase debieran tener algunos descriptores locales parecidos
- Con las imágenes de entrenamiento calcular un vocabulario visual global
- Para cada imagen calcular un vector global que resume la aparición de las palabras visuales
- Un clasificador buscará el patrón de palabras visuales que identifica cada clase

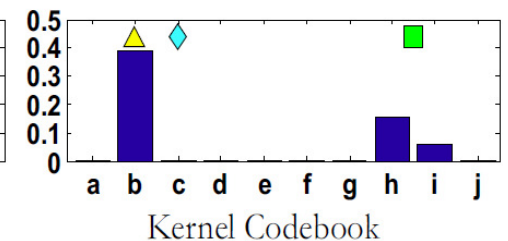
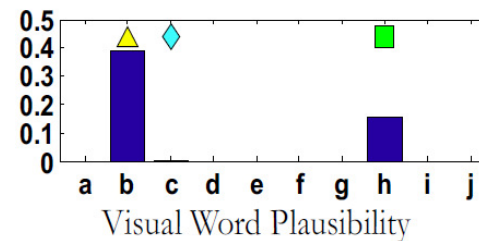
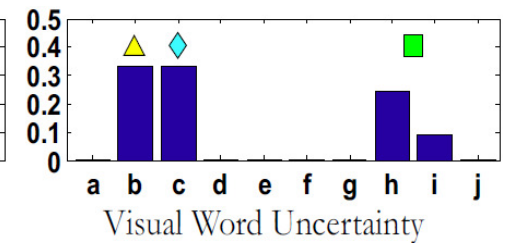
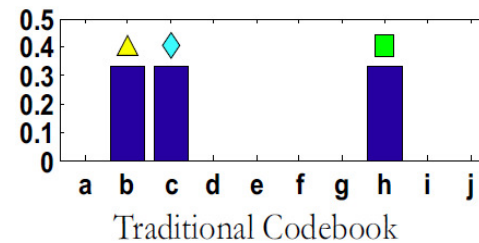
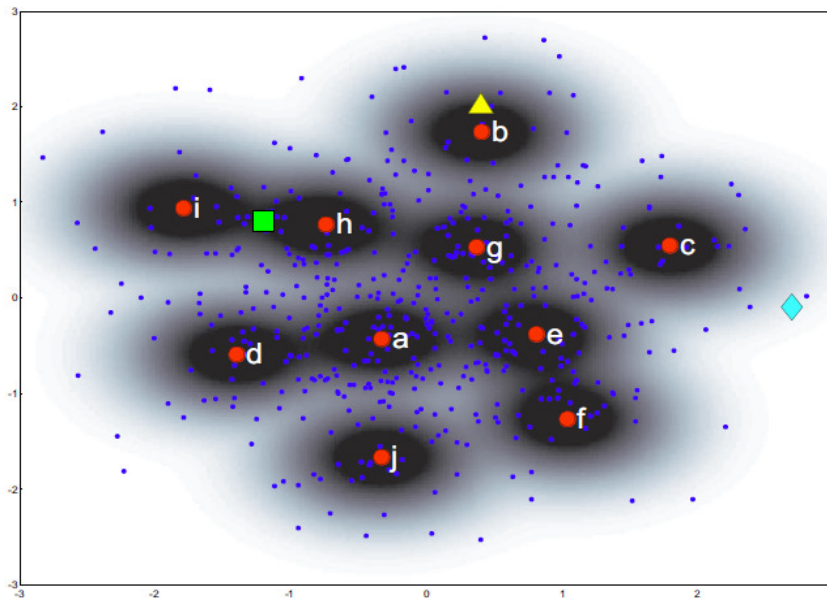


Clasificación de imágenes con BOVW

- Calcular Codebook:
 - Definir imágenes de entrenamiento I
 - Calcular conjunto L de descriptores locales de I
 - Algoritmo de clustering sobre L o una selección de L
 - Usualmente k-Means por su eficiencia
 - Notar que incluso selección aleatoria de centros (i.e. k-Means con 1 iteración) puede ser usado con una pequeña baja en efectividad
- Calcular el descriptor BOVW de cada imagen:
 - Calcular descriptores locales de la imagen
 - Reemplazar cada descriptor local por el codeword más cercano
 - Calcular el vector de frecuencias de cada codeword en cada imagen
- Entrenar un clasificador con los descriptores BOVW

Coding Difuso

- **Hard Assignment:** Cada descriptor local se redondea al codeword más cercano
- **Soft Assignment:** Ponderar cada asignación según distribuciones por codeword.



Van Gemert, Geusebroek. "Kernel Codebooks for Scene Categorization". 2008.

Coding/Pooling

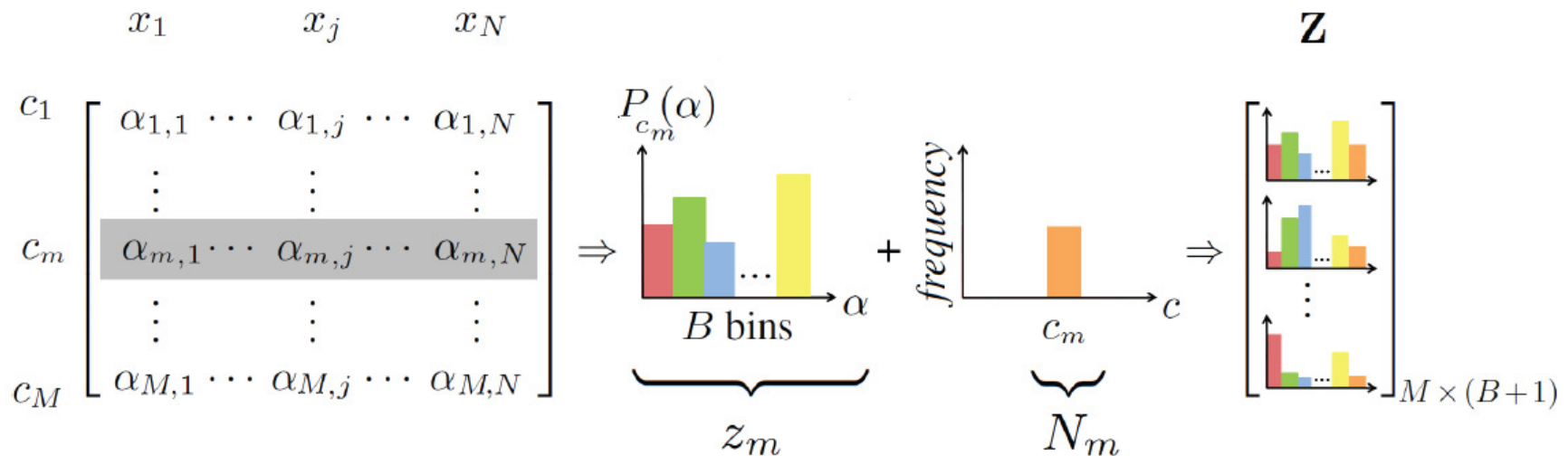
- Generalización para calcular el descriptor global de una imagen con descriptores locales $\{x_1, \dots, x_N\}$ usando un codebook $\{c_1, \dots, c_M\}$
- **Coding:** Para cada descriptor local x_j calcular los pesos de cada codeword: $\alpha_{1j}, \dots, \alpha_{Mj}$
 - Asignación al más cercano, asignación difusa.
- **Pooling:** Para cada codeword c_m calcular su peso global según todos sus pesos: $\alpha_{m1}, \dots, \alpha_{mN}$
 - Average-pooling, Max-pooling

$$\begin{array}{c}
 x_1 \qquad x_j \qquad x_N \\
 \begin{matrix} c_1 \\ \vdots \\ c_m \\ \vdots \\ c_M \end{matrix} \begin{bmatrix} \alpha_{1,1} & \cdots & \alpha_{1,j} & \cdots & \alpha_{1,N} \\ \vdots & & \vdots & & \vdots \\ \alpha_{m,1} & \cdots & \alpha_{m,j} & \cdots & \alpha_{m,N} \\ \vdots & & \vdots & & \vdots \\ \alpha_{M,1} & \cdots & \alpha_{M,j} & \cdots & \alpha_{M,N} \end{bmatrix} \Rightarrow g: \text{pooling}
 \end{array}$$

\Downarrow
 $f: \text{coding}$

Variantes

- **Bossa:** En la etapa de pooling, además del peso de cada c_i agregar un histograma de distancias normalizado de B bins (B entre 2 a 10)



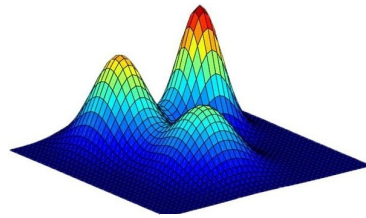


Vector of Locally Aggregated Descriptors (VLAD)

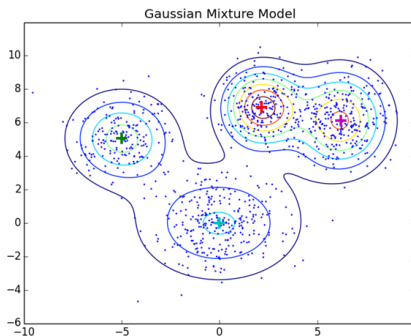
- Previo:
 - PCA sobre todos los descriptores locales, usar d dimensiones
 - Calcular codebook de tamaño M
- Etapa de Coding:
 - Para cada descriptor local se busca el codeword más cercano c_i
 - Calcular el “vector residual” $(x - c_i)$ para c_i (0 para el resto)
 - Mejora 1: calcular vector de redondeo normalizado $(x - c_i) / ||x - c_i||$
 - Mejora 2: utilizar matriz PCA independiente por c_i (pre-calculada)
- Etapa de Pooling
 - Para cada centro se suman sus vectores de redondeo
 - Se concatenan los vectores de error de los M codewords
- Descriptor de largo: $M \cdot d$
- “power-law normalization” del descriptor VLAD
 - $x_i = |x_i|^a \cdot \text{signo}(x_i)$ con $a < 1$ (Ej: raíz cuadrada)
- PCA sobre vectores VLAD

Fisher Vectors

- Uso de GMM para describir la distribución de descriptores locales de entrenamiento:
 - Combinación lineal de K gaussianas, cada una con un peso, una media y una matriz de covarianza



$$u_{\lambda}(x) = \sum_{k=1}^K w_k u_k(x)$$



$$u_k(x) = \frac{1}{(2\pi)^{D/2} |\Sigma_k|^{1/2}} \exp \left\{ -\frac{1}{2} (x - \mu_k)' \Sigma_k^{-1} (x - \mu_k) \right\}$$

$$\forall_k : w_k \geq 0, \quad \sum_{k=1}^K w_k = 1$$

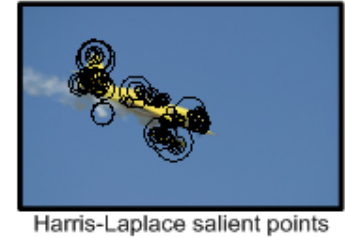


Fisher Vectors

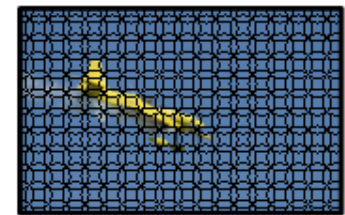
- Coding: Para cada descriptor local se debe calcular para cada una de las K gaussianas:
 - Probabilidad de haber sido generado (escalar)
 - Desviación de la media (vector d -dim)
 - Desviación de la varianza (vector d -dim)
 - Cada descriptor local se codifica por $2d+1$ valores por cada gaussiana
- Pooling:
 - Para cada gaussiana promediar las N codificaciones
 - Se concatenan los K codificaciones promedio
- Total: $K(2d+1)$
- Para acelerar cálculos usualmente se usan matrices Σ diagonales en la GMM

División Espacial

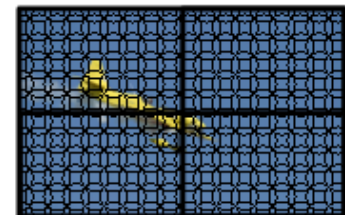
- Dense sampling:
 - No usar detector de keypoints
 - Los keypoints se obtienen a una tasa regular independiente de contenido de la imagen
 - Variar con un Δ constante la escala σ y ubicación (x,y)
- Spatial Pyramid:
 - Calcular un descriptor por cada zona 1x1, 2x2, 1x3, etc.
 - Concatenar los descriptores de todas las zonas



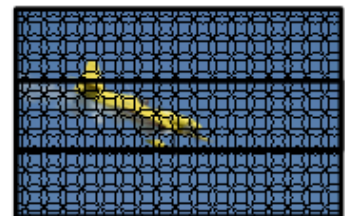
Harris-Laplace salient points



Dense sampling



Spatial pyramid (2x2)



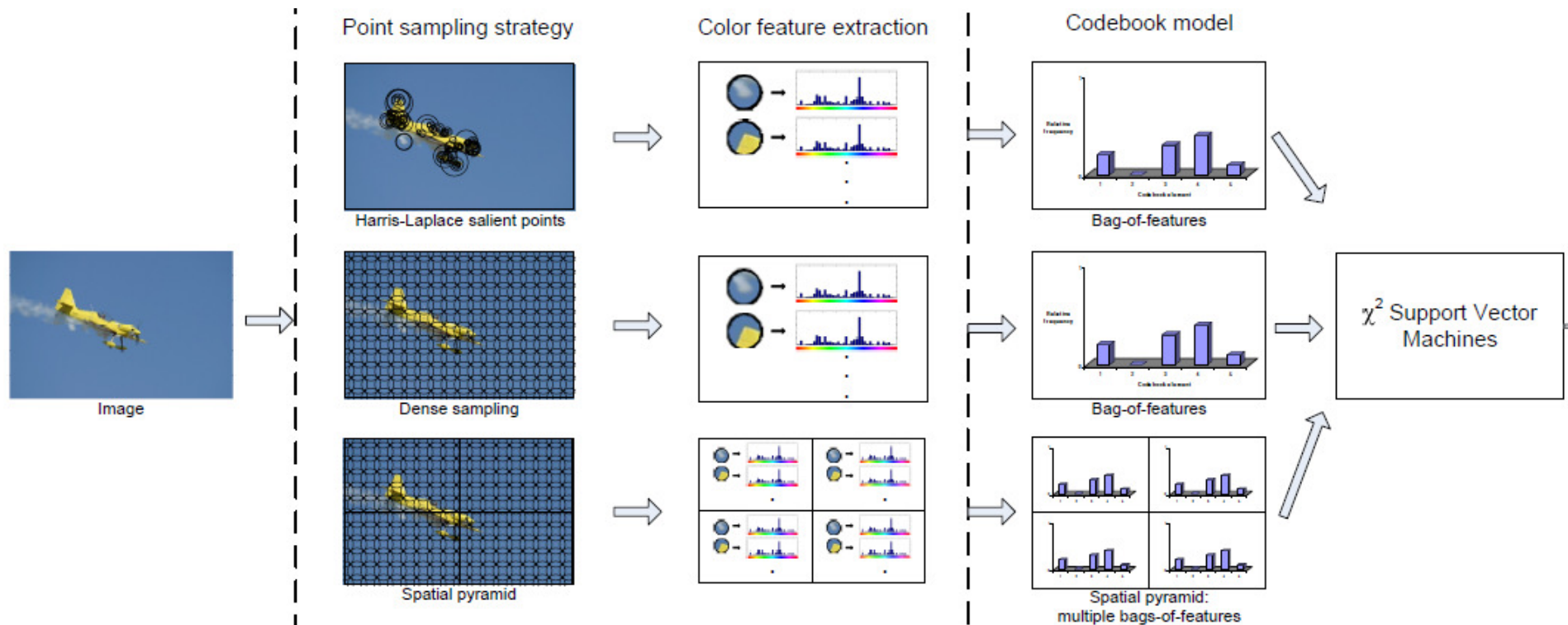
Spatial pyramid (1x3)



Combinación o Fusión

- Se desea crear un clasificador que combine N “modalidades” (e.g. color, audio, bordes, etc.).
- **Early fusion:** combinación en el espacio de descriptores.
 - Crear un solo descriptor como la concatenación de N descriptores.
 - Un único clasificador entrega una decisión combinada.
- **Late fusion:** combinación en el espacio semántico.
 - Se combina la decisión de N clasificadores.
 - Un clasificador por modalidad y luego se combinan los N scores (ya sea tomando el máximo, suma, etc.)

Clasificador de imágenes



Ver: <http://pascallin.ecs.soton.ac.uk/challenges/VOC/voc2008/workshop/tahir.pdf>



Bibliografía

- Tan, Steinbach, Kumar. “Introduction to Data Mining”. 2005.
 - <http://www-users.cs.umn.edu/~kumar/dmbook/index.php>
 - Cap 4, Cap 8.
- Szeliski. Computer Vision: Algorithms and Applications.
 - Cap 14.3, 14.4.
- Duda, Hart, Stork. “Pattern Classification”, second edition. 2000.
 - Cap 1.
- Camastra, Vinciarelli. “Machine Learning for Audio, Image and Video Analysis: Theory and Applications”. 2007.



Papers

- Sivic, Zisserman. “Video Google: A Text Retrieval Approach to Object Matching in Videos”. ICCV, 2003.
- van Gemert, Geusebroek. “Kernel Codebooks for Scene Categorization”. 2008.
- Uijlings, Smeulders, Scha. “Real-time Bag of Words, Approximately”. CIVR, 2007.
- Snoek, Worring, Smeulders. “Early versus Late Fusion in Semantic Video Analysis”. 2005.
- Avila, Thome, Cord, Valle. “Bossa: Extended bow formalism for image classification”. 2011.
- Delhumeau, Gosselin, Jégou, Pérez. “Revisiting the VLAD image representation”. 2013.
- Sanchez, Perronnin, Mensink, Verbeek. “Image Classification with the Fisher Vector: Theory and Practice”. 2013.