Study Guide: Introduction and Applications of Features in Computer Vision

This guide provides a detailed overview of salient elements (features) in images, their detection, description, matching, and how they are applied in higher-level computer vision tasks like object recognition.

1. Introduction to Features (Salient Elements)

Definition: Features are **"meaningful" parts of an image** that serve as fundamental elements in computer vision activities. They are crucial for tasks such as determining if two images depict the same subject or identifying common/different elements within them.

Main Components of Feature Processing:

- **Feature Detection**: The process of **finding a "stable" (easily detectable) point** in an image, often referred to as a **keypoint**.
- **Feature Description**: Creating a **description of the area surrounding the detected keypoint**, also known as a "signature" or "fingerprint". This description is typically a vector representation.
- **Feature Matching**: The task of **evaluating features in two images to find similar ones** (good matches). Similarity is measured using a distance metric applied to the feature descriptors.

Output: The combined result of these processes is a **set of points along with the description of their respective regions**.

Ideal Keypoints (Wishlist):

- **Stable and Repeatable**: Should be consistently detected across different views or conditions. Stability is measured by a repeatability score, which is the ratio of point-to-point correspondences that can be established to the minimum number of keypoints in the two images.
- **Invariant to Transformations**: Should remain detectable even if the image undergoes transformations like rotations.
- **Insensitive to Illumination Changes**: Brightness variations should not prevent their detection.
- **Accurate**: Precisely localized within the image.

Ideal Descriptor (Wishlist):

- **Robust**: Resistant to occlusion, clutter, noise, blur, compression, and discretization.
- **Discriminative**: Capable of distinguishing between different features effectively.
- **Stable**: Consistent despite changes in viewing angle and illumination.
- **Computationally Efficient**: Able to be calculated quickly, especially when many features are present in an image.

Applications of Features: Features are foundational for several computer vision systems, including:

- **Motion Detection**
- **Image Stitching** (creating mosaics from multiple images)
- **3D Reconstruction** and **Structure from Motion (SfM)**
- **Instance Matching/Object Localization**
- **Query by Example, Place Recognition**
- **Object Detection** (using a set of features)
- **Tracking** (following patterns in video streams)

2. Feature Detection: Corner and Blob Detectors

Feature detection, or keypoint detection, focuses on identifying salient points in an image. Unlike uniform regions or simple edges, **corner points** are particularly good candidates because they produce a large difference when a small image patch is shifted in any direction.

Harris Corner Detector:
One of the earliest and most well-known corner detection algorithms.

- **Intuition**: A patch on a uniform region shows small differences when shifted; on an edge, large differences occur only when shifted perpendicular to the edge; but on a corner, large differences occur regardless of the shift direction.
- **Similarity Measurement**: Uses an **auto-correlation function**, `E(Δx, Δy)`, which quantifies the change in the patch for a given displacement.
- **Auto-correlation Matrix (A) or Harris Matrix (M)**: Derived by approximating `E` using a Taylor series. This matrix describes how the region changes for a small displacement.
    - **Eigenvalues of A**: Analyzing the eigenvalues provides information about the patch type:
        - **Both eigenvalues are small**: Uniform region.
        - **Only one large eigenvalue**: Edge.
        - **Both eigenvalues are large**: Corner.
- **Keypoint Selection Criteria**: Harris's formula `det(A) - α * trace(A)^2` is commonly used, which is equivalent to `λ0λ1 - α(λ0+λ1)^2`.
- **Invariance**: The Harris detector is **invariant to brightness offset, shifts, and rotations**, but it is **not invariant to scaling**.

USAN/SUSAN Corner Detector:
An alternative to Harris that **analyzes a circular window around the point without using derivatives**.

- **USAN (Univalue Segment Assimilating Nucleus)**: Defined as the portion of the circular window whose intensity differs from the central pixel (nucleus) within a given threshold.
- **SUSAN (Smallest USAN)**: Detects corners when the USAN area is small, edges when it's about half the mask area, and uniform regions when it's similar to the mask area. It is robust to noise and can detect both edges and corners.

Blob Features:
Instead of sharp points, blobs are regions where properties (e.g., intensity) are approximately constant internally but distinct from the surrounding areas.

MSER (Maximally Stable Extremal Regions):
MSERs are connected areas characterized by nearly uniform intensity, surrounded by a contrasting background, effectively serving as a **blob detector**.

- **Algorithm**: Involves applying a series of thresholds to the image, computing connected binary regions, calculating statistics (like area or convexity) for each region, and analyzing the persistence (stability) of each blob.
  Le zone che rimangono quasi uguali anche quando cambi la soglia di luminosità sono considerate stabili → e quindi importanti.
- **"Extremal" Property**: All pixels within an MSER have intensities either consistently higher (bright regions) or consistently lower (dark regions) than all pixels on its outer boundary.

3. SIFT (Scale Invariant Feature Transform)

SIFT is a highly reliable and widely used algorithm for both keypoint detection and description.

Strong Points:

- **Local**: Robust to occlusions.
- **Distinctive**: Can differentiate objects in large databases.
- **Dense**: Capable of finding many features even on small objects.
- **Efficiency**: Relatively fast computation.
- **Key Invariances**: SIFT is specifically designed to be **invariant to scale and rotation**.

Phases of the SIFT Algorithm:

1. **Scale-Space Extrema Detection**:
   - **Scale Space**: Created by applying Gaussian smoothing (with varying `sigma`) to the image. It is organized into **octaves**, where the Gaussian `sigma` is doubled for each successive octave. Within an octave, intermediate scales are used.
   - **Difference of Gaussians (DoG)**: SIFT computes DoG images by subtracting consecutive layers (smoothed images) within the scale space $D(x, y, \sigma) = L(x, y, k\sigma) - L(x, y, \sigma)$. The DoG filter is a **good approximation of the Laplacian of Gaussian (LoG)** filter, which is effective for edge detection and noise removal.
   - **Extrema Search**: Maxima and minima of the DoG are sought by comparing each pixel with its 8 neighbors in the current, previous, and next scale levels. This comprehensive comparison across scales ensures **scale independence**.
     Per ogni pixel in ogni scala, si controllano i 26 pixel vicini (8 nello stesso livello, 9 sopra, 9 sotto). Se il pixel è un massimo o minimo locale, è un keypoint candidato.
2. **Keypoint Localization**:
   - Refines the keypoint location to sub-pixel accuracy using **Taylor series expansion** and interpolation along x, y, and scale (sigma).
   - Utilizes the **Hessian matrix** (similar to the auto-correlation matrix in Harris corners) to discard points with low derivatives or that are edge points, ensuring only true

corners/blobs are selected. This involves ensuring both eigenvalues of the Hessian are large, indicating high curvature in all directions.

3. **Orientation Measurement**:
   - Achieves **rotation invariance** by assigning a dominant orientation to each keypoint.
   - The **gradient magnitude and orientation** are computed in the neighborhood of the keypoint using the Gaussian-smoothed image closest to the keypoint's scale.
   - An **histogram of gradient orientations** (36 bins, 10° each) is built around the keypoint.
   - The **dominant direction** is determined by the peak of this histogram, refined by fitting a parabola.
   - **Secondary peaks** (if their value is > 80% of the highest peak) are also considered, leading to the creation of duplicate keypoints, each with a different orientation, further enhancing robustness.
4. **Descriptor Calculation**:
   - For each keypoint, a descriptor is evaluated based on the image rotated according to the keypoint's measured orientation.
   - Within a **16x16 neighborhood** around the keypoint, *gradient* magnitudes and orientations are computed. These values are weighted by a Gaussian function centered on the keypoint.
   - This 16x16 region is divided into **16 sub-regions of 4x4 pixels**. For each 4x4 region, an **8-bin histogram of gradient orientations** (45° per bin) is calculated.
   - These 16 histograms are concatenated to form the final **SIFT descriptor, which consists of 128 elements** (16 regions * 8 bins).
   - **Refinements for Illumination Invariance**: The feature vector is normalized to unit length, and elements are thresholded at 0.2 before final normalization to 1, which helps with non-linear illumination effects like camera saturation.

Fase	**Cosa fa**
Scale-space	Trova punti salienti a tutte le scale
DoG e massimi locali	Estrae punti stabili confrontando con i vicini
Taylor e filtraggio	Localizza il punto esatto, rimuove punti instabili
Orientamento	Calcola la direzione dominante per rendere il punto invariante
Descrittore SIFT	Riassume i gradienti locali in un vettore di 128 valori

4. Beyond SIFT: Other Feature Descriptors

While SIFT is highly effective, other descriptors have been developed, often aiming for improved speed or different properties.

- **PCA-SIFT (Principal Component Analysis - SIFT)**:
  - Integrates **Principal Component Analysis (PCA)**, a dimensionality reduction technique, into the SIFT algorithm.
  - **PCA's Role**: PCA projects data onto a lower-dimensional space, maximizing variance, which helps in creating more compact representations, reducing computational workload, and highlighting important information.
  - **Modification**: PCA-SIFT keeps the first three SIFT steps (detection, localization, orientation) identical but **modifies the descriptor calculation** (step 4).

- Instead of weighted histograms, PCA-SIFT concatenates horizontal and vertical gradients from a 41x41 patch into a long vector (3042 elements), normalizes it, and then reduces its dimensionality (e.g., from 3042 to 36) using PCA.

- **SURF (Speeded Up Robust Features)**:
  - Designed for **speed-up** by using fast approximations of the Hessian matrix for keypoint detection and descriptors using **integral images**.
  - **Integral Image**: Each pixel in an integral image stores the sum of all pixels in the rectangle from (0,0) to that pixel. This allows for constant-time calculation of sums over any rectangular region, irrespective of its size.
  - **Keypoint Detection**: Approximates the Laplacian (LoG) using box filters (balck and white rectangles) and finds maxima of the Hessian matrix determinant in scale space. SURF generates a scale space by using filters of increasing size, where the computational time does not depend on filter size due to integral images, unlike SIFT which smooths and subsamples.
  - **Orientation**: Uses **Haar wavelets** to evaluate gradients in x and y within a circular neighborhood. Responses are smoothed with a Gaussian, plotted in a 2D space, and the dominant orientation is determined by summing horizontal and vertical responses, quantized into 60° bins.
  - **Descriptor**: Similar to SIFT, it considers 16 regions. For each region, it sums the horizontal and vertical *gradient responses* (dx *and* dy) and their absolute values (|dx| *and* |dy|). This results in 4 elements per region, leading to a **64-element feature size** (4x16).
  - **Comparison to SIFT**: SURF is **faster** but generally **less robust to illumination and viewpoint changes** than SIFT. Both are patented.

- **GLOH (Gradient Location-Orientation Histogram)**:
  - Another modification of the SIFT algorithm's descriptor calculation.
  - Computes a SIFT-like descriptor but uses a **log-polar location grid**.
  - It uses 3 radial bins and 8 angular bins (with the central bin not subdivided), combined with 16 orientations, yielding 272 orientation bins. PCA is then used to reduce its dimensionality, typically to 128 elements.

- **Binary Features**: These descriptors often aim for even greater speed and compactness by producing binary vectors, enabling very fast matching using Hamming distance.
  - **LBP (Local Binary Pattern)**:
    Originally for texture recognition.
    - Compares a central pixel with its surrounding neighbors (e.g., 8 neighbors in a circle). If a neighbor's intensity is greater than the center, it's assigned '1', otherwise '0', forming a binary sequence.
    - Histograms of these binary patterns are computed, optionally normalized. An entire image can be analyzed by dividing it into sub-windows and concatenating their histograms.

- **BRIEF (Binary Robust Independent Elementary Features)**:
  - Based on Gaussian smoothing and **comparing pairs of pixels** within a window.
  - A binary vector is built where each element is '1' if the first pixel in a pair is less than the second, and '0' otherwise.
  - The sampling pattern for pixel pairs can be fixed (e.g., 128, 256, or 512 pairs) or random, with isotropic Gaussian distribution often providing the best performance.
  - BRIEF vectors are compared using the **Hamming distance** (XOR operation).

1) Prendi una patch attorno al keypoint
2) Fai confronti di intensità tra coppie di pixel nella patch
3) Ripeti per n coppie → ottieni un vettore binario

- **ORB (Oriented FAST and Rotated BRIEF)**:
  - Combines the **FAST corner detector** with the BRIEF descriptor.
  - Adds **rotation invariance to BRIEF** by assigning an orientation to the keypoint based on the intensity centroid relative to the central pixel.

5. Feature Matching

Feature matching is a core task in computer vision, involving finding similar feature descriptors between images.

Matching Process:
- **Similarity Measurement**: Typically, **Euclidean distance** is used for dense, floating-point descriptors like SIFT, while **Hamming distance** is used for binary descriptors like BRIEF.
- **Matching Strategy**: The chosen strategy depends on the specific application (e.g., image stitching, object detection in clutter).

  - **Strategy 1: Maximum Distance**: Check al the descriptor. Matches features within a certain geometric distance. However, setting an appropriate threshold can be challenging.

  - **Strategy 2: Nearest Neighbor (NN)**: Considers only the nearest neighbor in the feature space. A threshold can still be applied to filter out poor matches.

  - **Strategy 3: Nearest Neighbor Distance Ratio (NNDR)**: A common approach where the ratio of the nearest distance (`d1`) to the second nearest distance (`d2`) is calculated (`NNDR = d1 / d2`). This helps to identify more distinctive matches.

Matching Performance Metrics:
- **True Positives (TP)**: Correctly identified matches.
- **True Negatives (TN)**: Correctly identified non-matches.
- **False Positives (FP)**: Non-matches wrongly identified as matches.
- **False Negatives (FN)**: Actual matches that were wrongly missed.
- **Precision**: `TP / (TP + FP)` – measures the accuracy of positive predictions.

- **Recall**: `TP / (TP + FN)` – measures the completeness of positive predictions.

6. Object Recognition and High-Level Vision

Object recognition is a broad field of computer vision focused on identifying objects in images or videos, moving towards extracting higher-level information. This can range from simply outputting a label to providing bounding boxes or segmented areas for multiple objects.

Challenges in Object Recognition: Systems must cope with various factors:
- **Different Camera Positions**
- **Perspective Deformations**
- **Illumination Changes**
- **Intra-Class Variations** (differences within the same object category, e.g., different types of chairs)

Approaches to Object Recognition:

Viola-Jones Face Detector (Revisited):
A widely used and **highly effective approach for fast object detection**, particularly for faces, but applicable to other targets.
- **Sliding Window Approach**: A window is moved across the image at multiple locations and scales to find matching regions. This requires fast processing for the huge number of non-face candidates.
- **Key Elements**:
  - **Haar Features**: Rectangular filters that subtract the sum of pixels in white areas from black areas. They are coarse but computationally efficient, especially when used with **integral images** (which allow constant-time computation). A 24x24 patch can have 160,000 possible Haar features.
  - **Weak Learners**: Simple classifiers that set a threshold on a single Haar feature to separate positive (face) and negative (non-face) examples, performing slightly better than a random guess.
  - **Boosting (AdaBoost)**: Combines many weak learners into a **strong classifier** via a weighted sum. AdaBoost iteratively selects the best weak classifier and reweights examples, giving more emphasis to misclassified samples.
  - **Cascading of Classifiers**: Organizes multiple strong classifiers into sequential **stages**. Each stage acts as a filter: if a sub-window is rejected by an early stage, it prevents subsequent, more complex stages from processing it, leading to **fast rejection of non-matching windows**. False negatives cause failure, but high false positive rates are acceptable at early stages. Viola-Jones used 38 stages with 6061 features.

Template Matching (TM):

- **Concept**: Finding instances of a pre-defined "template" (a model or representative instance) within an image.
- **Application**: A similarity measure is chosen to evaluate the match between the template and image regions.
- **Sliding Window**: The template is placed at every possible position across the image.
- **Comparison Methods**: Can compare pixel values, features, or edge/gradient orientations.
- **Similarity Metrics**:
  - **Sum of Squared Differences (SSD)**: `Σ(I - T)^2`.
  - **Sum of Absolute Differences (SAD)**: `Σ|I - T|`.
  - **Zero-mean Normalized Cross-Correlation (ZNCC)**: `(Σ (I - avg(I)) (T - avg(T))) / (std(I) * std(T))`. ZNCC is particularly useful for dealing with illumination changes as it subtracts the uniform illumination component.
- **Coping with Weak Points**: Scale changes can be handled by matching with several scaled versions of the template or resizing the image. Rotations can be managed by matching with multiple rotated versions of the template.
- **Generalized Hough Transform**: Can be viewed as a form of template matching for more complex shapes defined by a general equation `g(v, c) = 0`.

1) **Histogram of Oriented Gradients (HOG)**:

- **Approach**: Works by sliding a window, characterizing its content by evaluating edge magnitude and phase to produce a descriptor, and then classifying this descriptor.
- **Descriptor Evaluation Steps**:
  1. Intensity normalization/histogram equalization and smoothing.
  2. Calculation of the edge map (magnitude and phase).
  3. Evaluation of **edge histograms on non-overlapping 8x8 cells** (e.g., 9 bins of 20° covering 180° for each cell).
  4. Creation of **overlapping blocks of 2x2 cells**.
  5. **Normalization of voting vectors** over each block and creation of block vectors (e.g., 36 elements per block).
  6. Serialization of block vectors to form the final HOG descriptor.
- **Characteristics**: The HOG descriptor characterizes the content of a bounding box and typically requires bounding boxes to be normalized to a standard size. Multiple scales can be handled by resizing bounding boxes of different dimensions to this standard size. It is commonly used to train a classifier. For a 64x128 window, it results in 8x16 cells, 7x15 blocks, and a feature size of 3780 elements.

2) **Bag of Words (BoW)**:

- **Concept**: An approach adapted from document analysis for **image and object classification**, designed to be invariant to factors like viewpoint and deformations. It decomposes complex patterns into (semi) independent "visual words".

Pipeline:

1. **Codebook Generation**:
    - **Extract features** (keypoints and descriptors, e.g., SIFT) from a set of sample images of the object(s) to be recognized. Multiple features are collected, potentially including similar ones. This generates data points in the feature space (e.g., 128 dimensions for SIFT).
    - Perform **clustering** (e.g., K-means) on these collected features in the feature space. This groups similar elements seen in single or multiple samples.
    - Each cluster generates a **representative sample (e.g., centroid)**, which becomes a "visual word" in the codebook. The codebook contains all words describing all elements to be recognized.
2. **Test Image Analysis**:

- **Extract features** from the test (query) image.
- **Compare and associate** these extracted features with the visual words in the generated codebook.
- Generate a **histogram (or frequency vector)** of the occurrences of each visual word found in the test image.

3. **Object Recognition/Image Classification**:

- The frequency vector from the test image is compared against the frequency vectors of known objects (e.g., using Euclidean distance or dot product).
- Alternatively, these frequency vectors can serve as input for an image classifier.