Feature detection and matching - Points and patches

Objectives

- Learn about keypoint features or interest points
- Learn how to handle patches of pixels surrounding the point location
- The techniques is used in Feature detectors
- The techniques is used in Feature descriptors
- The techniques is used in Feature matching
- The techniques is used in Feature tracking

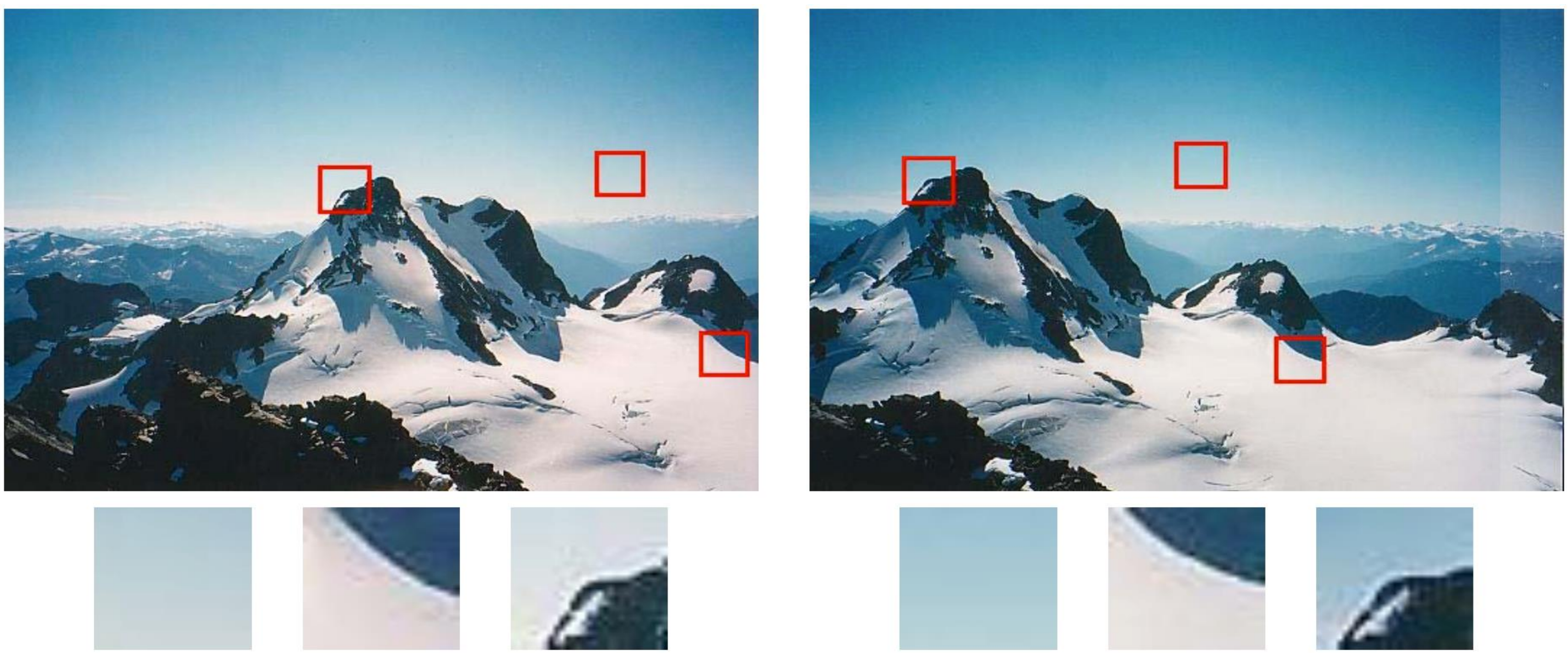
How to determine matched images?

Points, Patches and Feature

- Image patch as the name suggests is a group of pixels in an image(correspondence points)
- Feature is a piece of information about the content of an image.
- Features may be specific structures in the image such as points, edges, or objects.
- Point features can be used to find a sparse set of corresponding locations in different images
- Feature extraction involves reducing the number of resources required to describe a large set of data

Feature detectors

- How can we find image locations where we can reliably find correspondences with other images?
- What are good features to track?



Feature detectors

- Matching criterion for comparing two image patches:

$$E_{\mathrm{WSSD}}(\boldsymbol{u}) = \sum_i w(\boldsymbol{x}_i)[I_1(\boldsymbol{x}_i + \boldsymbol{u}) - I_0(\boldsymbol{x}_i)]^2$$

  – where I0 and I1 are the two images being compared, u = (u, v) is the displacement vector, w(x) is a spatially varying weighting function, and the summation i is over all the pixels in the patch.

- Auto-correlation function

$$\begin{aligned} E_{\mathrm{AC}}(\Delta\boldsymbol{u}) &= \sum_i w(\boldsymbol{x}_i)[I_0(\boldsymbol{x}_i + \Delta\boldsymbol{u}) - I_0(\boldsymbol{x}_i)]^2 \\ &\approx \sum_i w(\boldsymbol{x}_i)[I_0(\boldsymbol{x}_i) + \nabla I_0(\boldsymbol{x}_i)\cdot\Delta\boldsymbol{u} - I_0(\boldsymbol{x}_i)]^2 \\ &= \sum_i w(\boldsymbol{x}_i)[\nabla I_0(\boldsymbol{x}_i)\cdot\Delta\boldsymbol{u}]^2 \\ &= \Delta\boldsymbol{u}^T \boldsymbol{A} \Delta\boldsymbol{u}, \end{aligned}$$

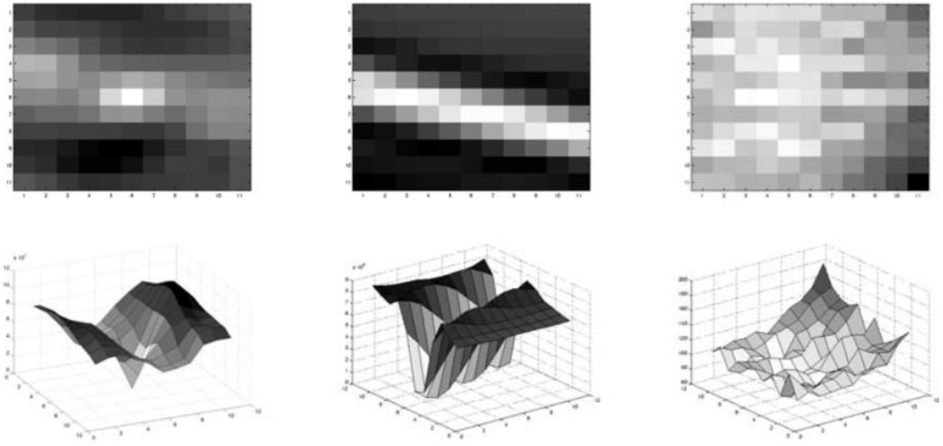
$$\nabla I_0(\boldsymbol{x}_i) = (\frac{\partial I_0}{\partial x}, \frac{\partial I_0}{\partial y})(\boldsymbol{x}_i)$$

  – Auto-correlation matrix

$$\boldsymbol{A} = w * \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix}$$

Feature detectors

(a)



(b) (c) (d)

Feature detectors

- The basic auto-correlation-based keypoint detector algorithm
  - Compute the horizontal and vertical derivatives of the image Ix and Iy by convolving the original image with derivatives of Gaussians
  - Compute the three images corresponding to the outer products of these gradients
  - Convolve each of these images with a larger Gaussian.
  - Compute a scalar interest measure using one of the formulas discussed above
  - Find local maxima above a certain threshold and report them as detected feature point locations.

Feature detectors - Harris Corner Detector

- It basically finds the difference in intensity for a displacement of (u,v) in all directions
  - We have to maximize this function E(u,v) for corner detection

$$E(u,v) = \sum_{x,y} \underbrace{w(x,y)}_{\text{window function}} \, [\underbrace{I(x+u,y+v)}_{\text{shifted intensity}} - \underbrace{I(x,y)}_{\text{intensity}}]^2$$

- Where $E(u,v) \approx \begin{bmatrix} u & v \end{bmatrix} M \begin{bmatrix} u \\ v \end{bmatrix}$ and Ix and Iy are image derivatives in x and y directions respectively.
- Compute score which determines if a window can contain a corner or not.

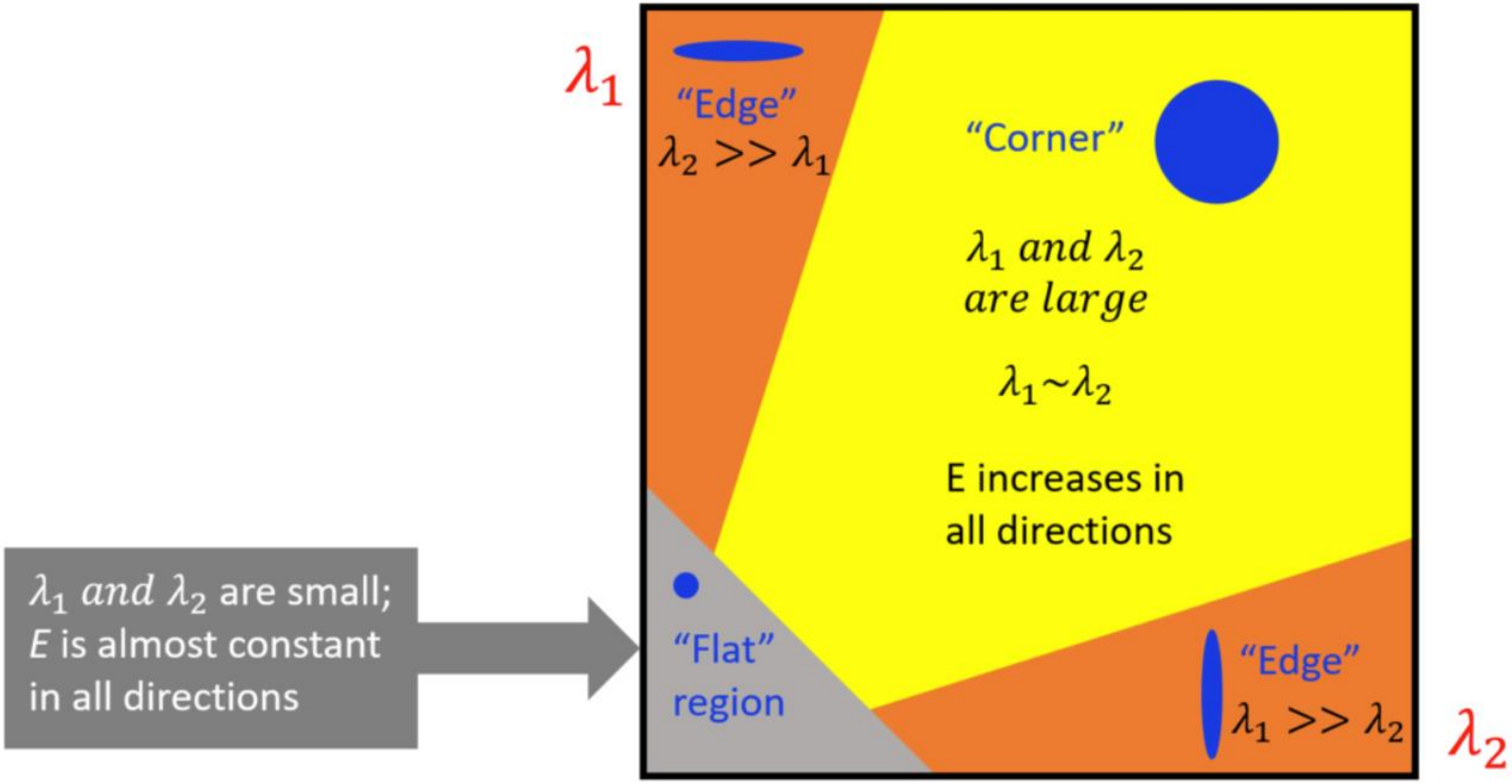
$$M = \sum_{x,y} w(x,y) \begin{bmatrix} I_x I_x & I_x I_y \\ I_x I_y & I_y I_y \end{bmatrix} \qquad R = \det(M) - k(\operatorname{trace}(M))^2$$

- Where det(M)=λ1λ2, trace(M)=λ1+λ2, λ1 and λ2 are the eigenvalues of M

Feature detectors - Harris Corner Detector

- The magnitudes of these eigenvalues decide whether a region is a corner, an edge, or flat
  - When $|R|$ is small, which happens when $\lambda 1$ and $\lambda 2$ are small, the region is flat.
  - When $R<0$, which happens when $\lambda 1 >> \lambda 2$ or vice versa, the region is edge.
  - When R is large, which happens when $\lambda 1$ and $\lambda 2$ are large and $\lambda 1 \sim \lambda 2$, the region is a corner.

Feature detectors - Harris Corner Detector

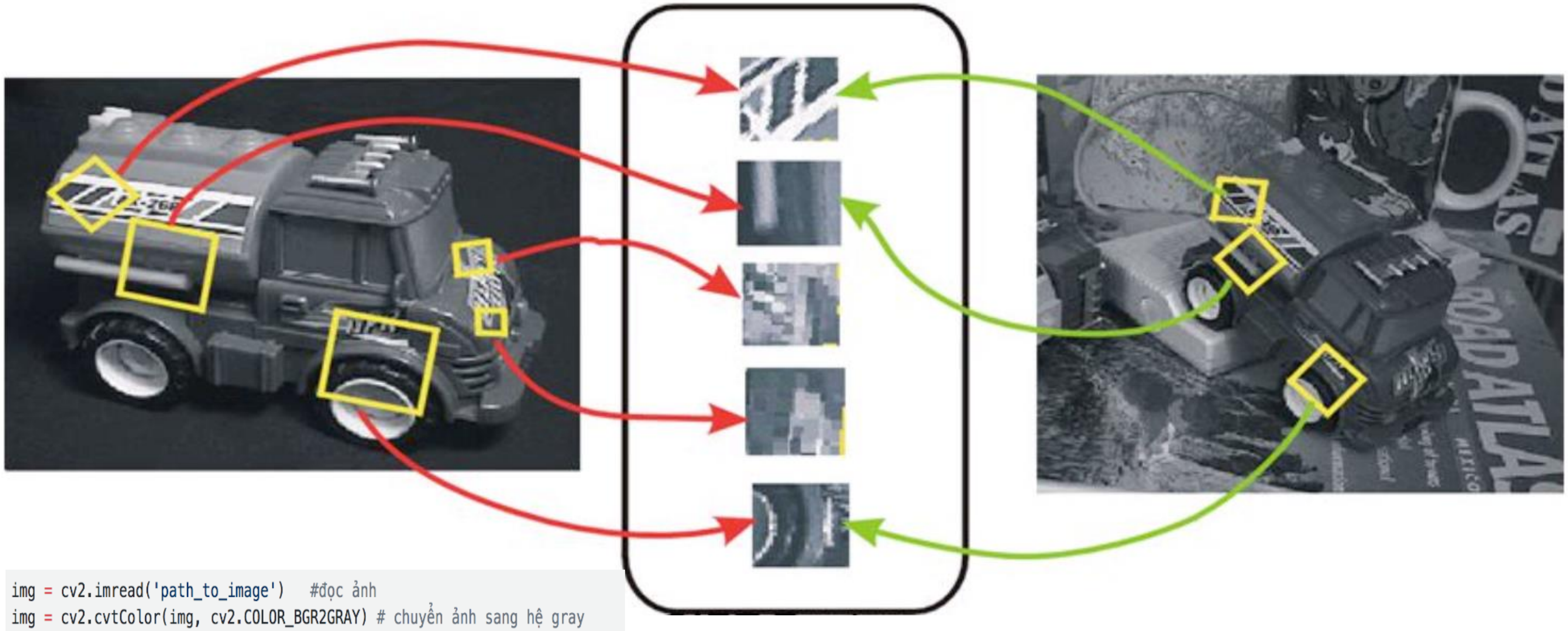


Feature detectors - Harris Corner Detector

Feature detectors - Scale-Invariant Feature Transform

- It is a feature detection algorithm in computer vision to detect and describe local features in images
- SIFT Algorithm :
  - Scale-space peak selection: Potential location for finding features.
  - Keypoint Localization: Accurately locating the feature keypoints.
  - Orientation Assignment: Assigning orientation to keypoints.
  - Keypoint descriptor: Describing the keypoints as a high dimensional vector.
  - Keypoint Matching

Feature detectors - Scale-Invariant Feature Transform



```python
img = cv2.imread('path_to_image')   #đọc ảnh
img = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY) # chuyển ảnh sang hệ gray

sift = cv2.xfeatures2d.SIFT_create() #khởi tạo đối tượng sift

kp, des = sift.detectAndCompute(img,None)   #Đối tượng này có phương thức
print(des.shape)

img=cv2.drawKeypoints(gray,kp,img)
cv2.imwrite('path_to_image',img) #lưu ảnh
```

Feature detectors - Others

- Speeded-Up Robust Features (SURF) — This is a faster version of SIFT as the name says.
- Features from Accelerated Segment Test (FAST) — This is a much more faster corner detection technique compared to SURF.
- Binary Robust Independent Elementary Features (BRIEF) —This technique reduces the memory usage by converting descriptors in floating point numbers to binary strings.
- Oriented FAST and Rotated BRIEF (ORB) — uses FAST keypoint detector and BRIEF descriptor.

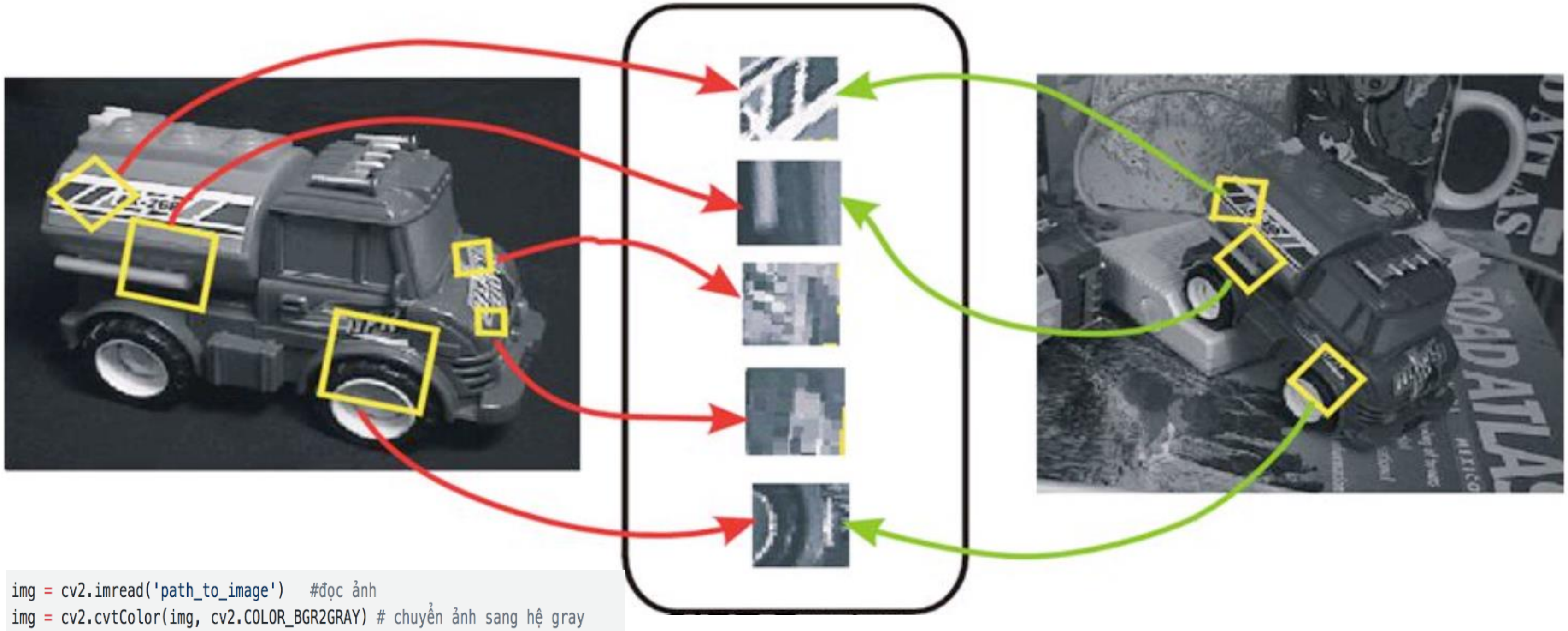
Feature descriptors

- After detecting features (keypoints), you must determine which features come from corresponding locations in different images.
- A feature descriptor is an algorithm that takes an image and outputs feature descriptors/feature vectors.
- Feature descriptors encode interesting information into a series of numbers and act as a sort of numerical "fingerprint" that can be used to differentiate one feature from another.
- Ideally, this information would be invariant under image transformation, so we can find the feature again even if the image is transformed in some way

Feature descriptors - Scale-Invariant Feature Transform

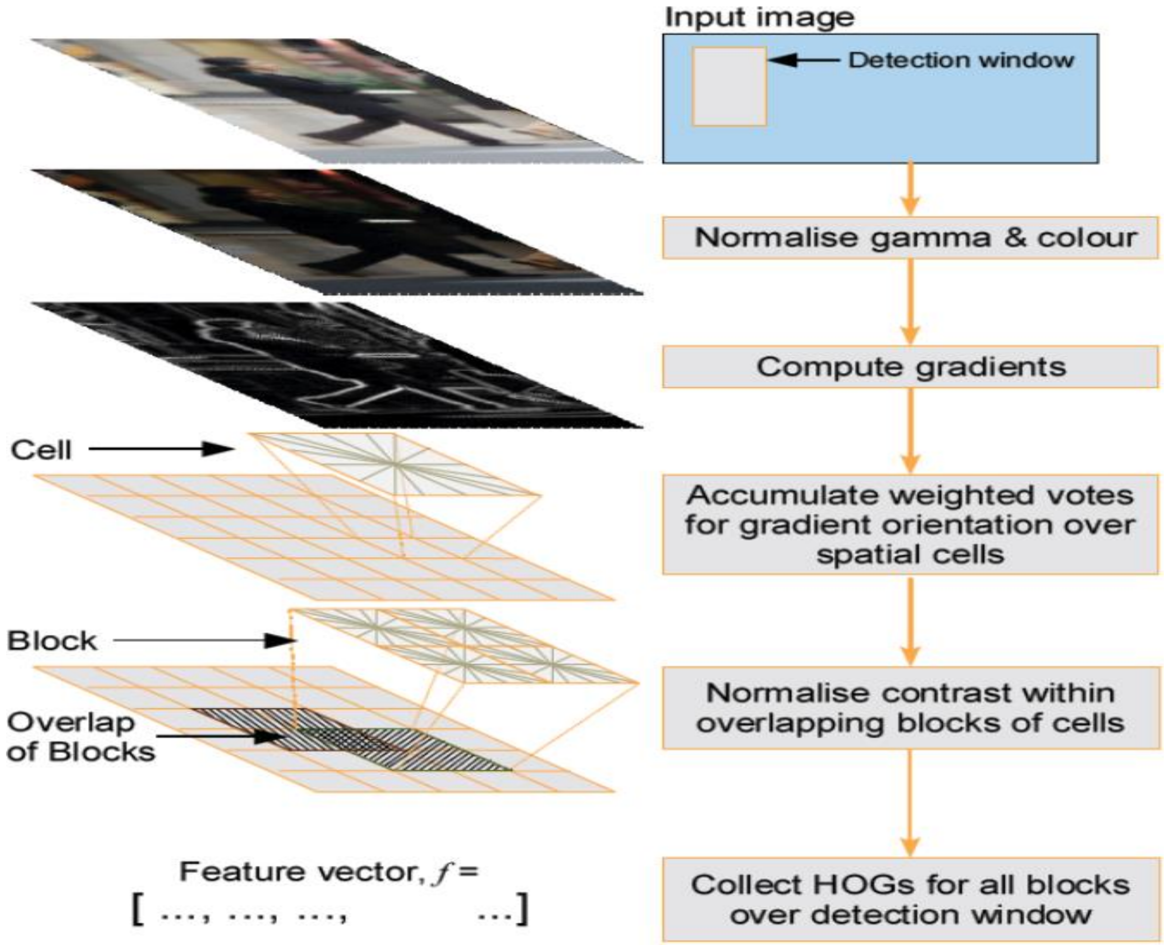
- It is a feature detection algorithm in computer vision to detect and describe local features in images
- SIFT Algorithm :
  - Scale-space peak selection: Potential location for finding features.
  - Keypoint Localization: Accurately locating the feature keypoints.
  - Orientation Assignment: Assigning orientation to keypoints.
  - Keypoint descriptor: Describing the keypoints as a high dimensional vector.
  - Keypoint Matching

Feature descriptors
- Scale-Invariant Feature Transform

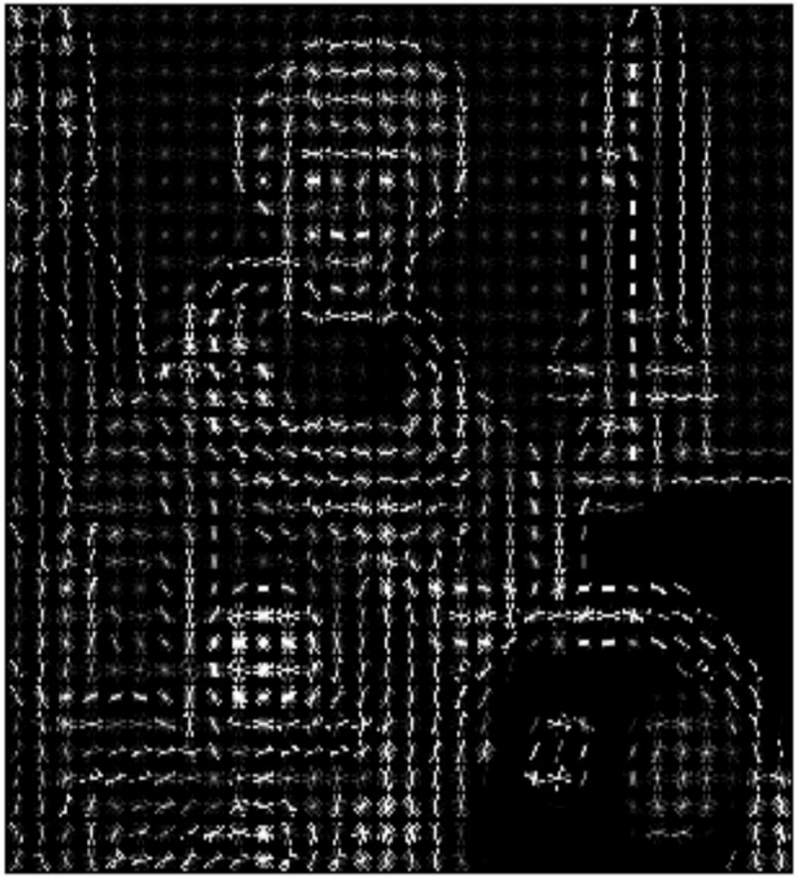


```
img = cv2.imread('path_to_image')   #đọc ảnh
img = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY) # chuyển ảnh sang hệ gray

sift = cv2.xfeatures2d.SIFT_create() #khởi tạo đối tượng sift

kp, des = sift.detectAndCompute(img,None)   #Đối tượng này có phương thức
print(des.shape)

img=cv2.drawKeypoints(gray,kp,img)
cv2.imwrite('path_to_image',img) #lưu ảnh
```

Feature descriptors
- Histogram of Oriented Gradients



Feature descriptors
- Histogram of Oriented Gradients



Feature descriptors - Others

- Steerable filters: is an orientation-selective convolution kernel used for image enhancement and feature extraction that can be expressed via a linear combination of a small set of rotated versions of itself
- Shape context and geometric blur: Geometric blur is simply an average over geometric transformations of a signal. This turns out to be a useful operation for comparing two signals when some geometric distortion is expected.

Feature matching

- After extracted features and their descriptors from two or more images→ establish some preliminary feature matches between these images.
- The algorithm is based on comparing and analyzing point correspondences between the reference image and the target image
- Feature matching steps:
  - Select a matching strategy, which determines which correspondences are passed on to the next stage for further processing.
  - Devise efficient data structures and algorithms to perform this matching as quickly as possible.

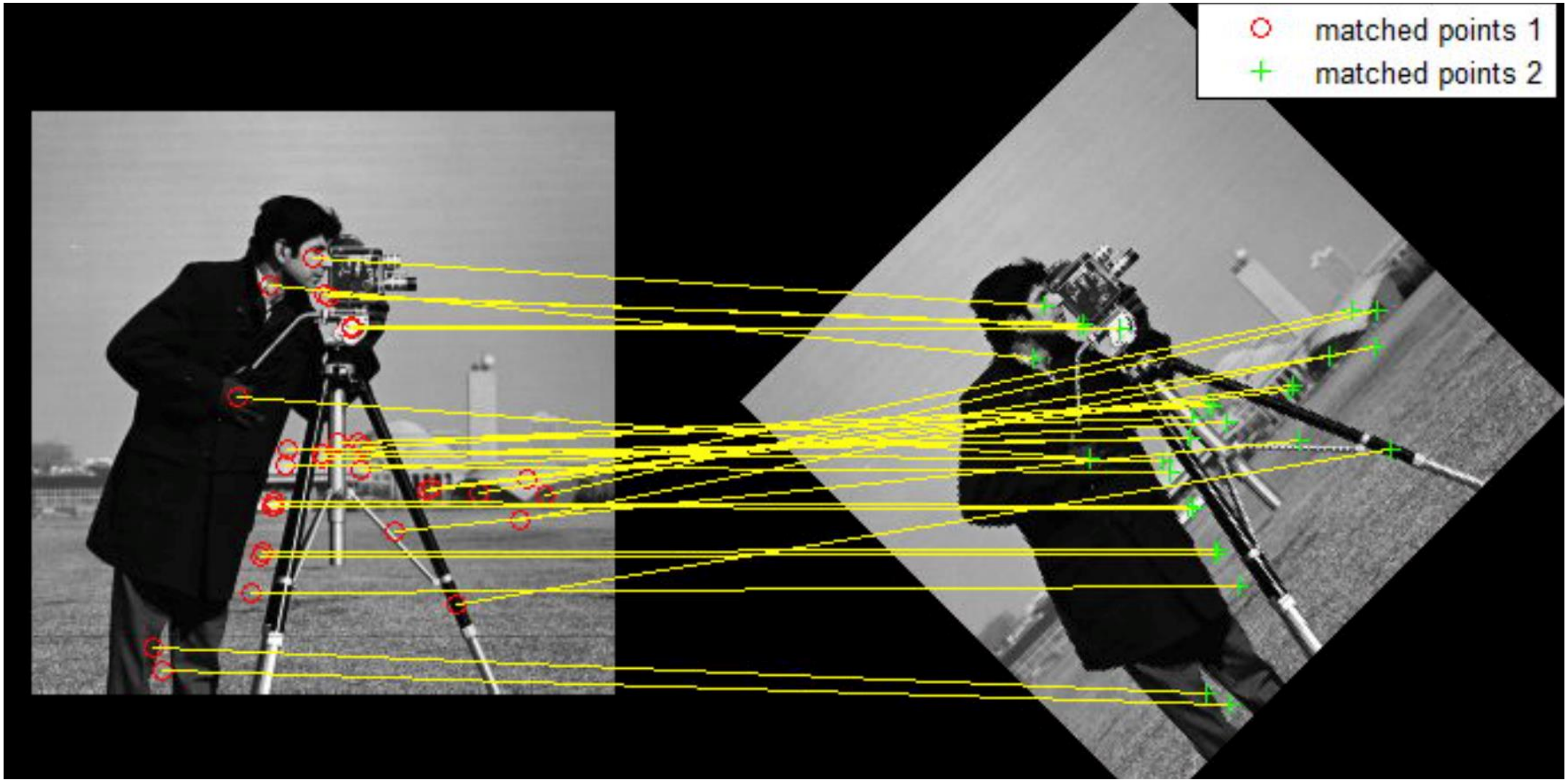
Feature matching- Matching strategy and error rates

- Determining which feature matches are reasonable to process further depends on the context in which the matching is being performed.
  - Euclidean (vector magnitude) distances in feature space can be directly used for ranking potential matches.
  - The simplest matching strategy is to set a threshold (maximum distance) and to return all matches from other images within this threshold.
    - Threshold too high results in too many false positives--> incorrect matches being returned.
    - Threshold too low results in too many false negatives--> too many correct matches being missed

Feature matching - Efficient matching

- Once we have decided on a matching strategy, you still need to search efficiently for potential candidates.
- The simplest way to find all corresponding feature points is to compare all features against all other features in each pair of potentially matching images
  - Multi-dimensional search tree: k-d trees,
  - Multi-dimensional hashing: Haar wavelets, Locality sensitive hashing

Feature matching



Feature tracking

- The feature point tracking problem consists of detecting images of particles in a digital video sequence and linking these detections over time to follow the traces of individual particles.
- The feature tracking problem: considering two consecutive frames from a video sequence, visual features are extracted within the first frame. Then, it is tried to find the same features back in the next frame.
- The expected amount of motion and appearance deformation between adjacent frames is expected to be small.
- Searching for locations where the corresponding patch has low squared difference often works well enough.

Feature tracking

Summary

- Learn about keypoint features or interest points
- Learn how to handle patches of pixels surrounding the point location
- The techniques is used in Feature detectors
- The techniques is used in Feature descriptors
- The techniques is used in Feature matching
- The techniques is used in Feature tracking