Feature Extraction

What are features and what is their use?

- Feature is a piece of information that describes an image or a part of it
  - Corners
  - Edges
  - Circles, ellipses, lines, blobs etc

- Features are useful to match two images
  - Find common points in two images to stitch them
  - Object recognition, face recognition
  - Structure from motion, stereo correspondence

- **Why not match pixels values?** – Pixel values change with light intensity, colour and direction. They also change with camera orientation.

Properties of Good Features

- Repeatability
  - Should be detectable at the same locations in different images despite changes in viewpoint and illumination
- Saliency (descriptiveness)
  - Same points in different images should have similar features
  - And vice-versa
- Compactness (affects speed of matching)
  - Fewer features
  - Smaller features

Feature Extraction has Two Steps

1. Feature detection
   - Where to extract features
   - Not all locations are good to extract features

2. Feature extraction
   - Orientation of the extraction window
   - What to encode in the feature

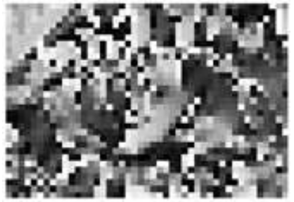
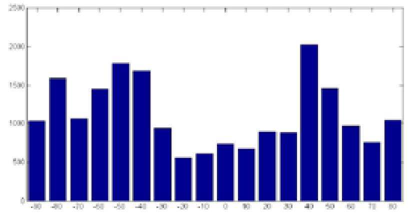
Examples of Feature Extraction

- Haar features
- Hough Transform/ Hough Features
- HOG: Histogram of Oriented Gradients
- LBP: Local Binary Patterns
- Harris Corners
- Scale Invariant Feature Transform (SIFT)

Histogram of Oriented Gradients (HOG)

- Calculate gradient magnitude and orientation at each pixel
- Divide orientations into N bins e.g. 360 degrees can be divided into 18 bins
- The gradient magnitude of each pixel is then voted into the bin corresponding to its orientation
- Improved variants
  - Distribute the gradient magnitude among neighboring bins (soft voting)
  - Divide image into cells, compute HOG for each cell and concatenate them
  - Use a Gaussian window to weight the gradient magnitudes
  - Normalize the final HOG to unit magnitude



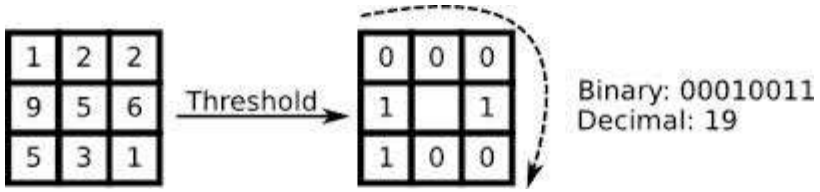



HOG Applications

- Can be used for detection of objects or actions
- Has shown very good results for pedestrian detection
- Has shown good results for action recognition

Local Binary Pattern (LBP)

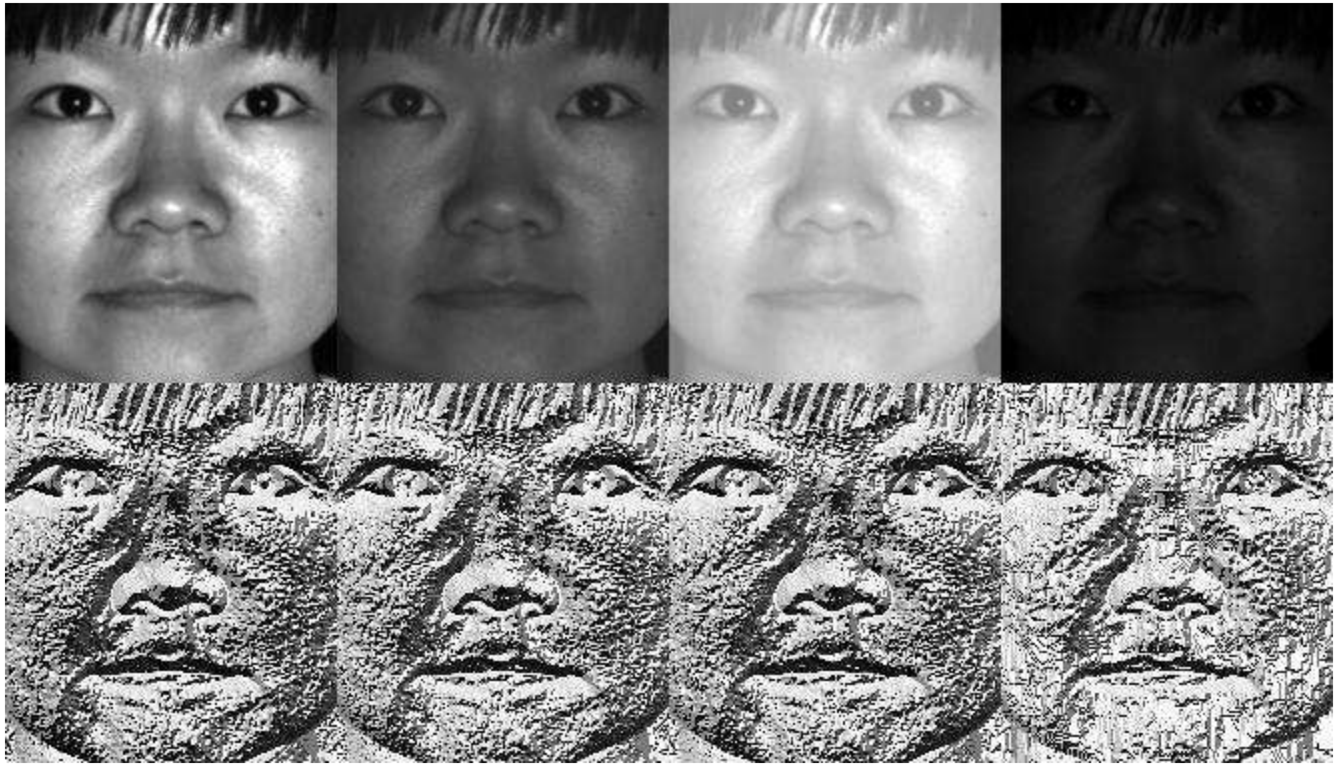
- Compare 8-connected neighborhood with center pixel
- If pixel > center, replace with ‘1’ else ‘0’
- Construct a binary number by going clockwise
- Replace the center pixel with the decimal value of the binary number



- Binary number is sensitive to starting point – LBP is not rotation invariant
- Rotate binary string to minimize decimal value for rotation invariance
- Minor changes in illumination can change the decimal value
- Partition image into cells and construct LBP histograms in each cell

LBP Example from OpenCV

- Notice how LBP feature are illumination invariant



Should we Extract Features from all Points?

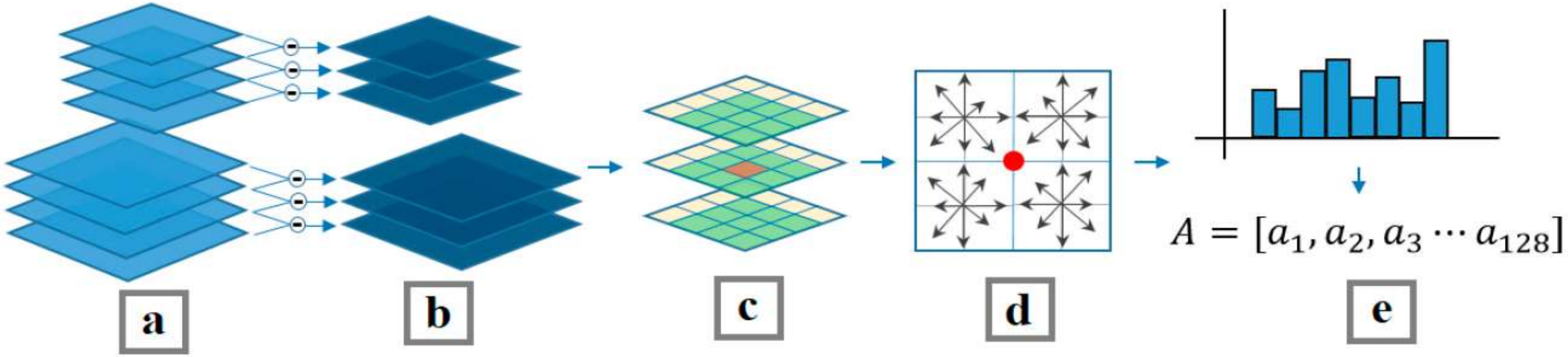
- LBP, HOG and Haar features do not have an embedded interest point detection

- Such features must be calculated at each pixel or globally from the overall image
  - Sensitive to occlusions
  - Sensitive to orientation
  - Computationally expensive

SIFT Feature Extraction

- The aim of SIFT is the extraction of features that are invariant to image rotation and scale
- The rotation invariance is ensured using the gradient orientations and magnitudes of the pixels around the key points, and the scale invariance is ensured using the scale space approach.
- The steps of the algorithm are
  - scale space production,
  - difference of Gaussian (DOG),
  - finding local maximum and minimum,
  - eliminating bad points,
  - assigning orientation to key points and
  - Feature transformation

The steps of SIFT feature extraction



(**a**) scale space generation; (**b**) DOG image generation; (**c**) detection of local maximum and minimum; (**d**) gradient calculation; (**e**) histogram calculation and generation of 128-dimensional vectors.

The steps of SIFT feature extraction

- The SIFT algorithm first generates the scale space images by convolving the input images using Gaussian Convolution and attains the differences between Gaussian (DOG) images by subtracting adjacent images. (See Fig (a) and (b) in the previous slide)
    - Construct image pyramids with different scales
    - Apply Gaussian convolutions with different sigma values on each image scales
- Local maximum and minimum points are then detected by comparing neighbour pixels with the target pixels in the current and adjacent DOG images. Local maximums and minimums are saved as key points (See Fig(c)).
- The key points, which lie on the edges and have low contrast, are eliminated.

The steps of SIFT feature extraction

- Gradient orientations and magnitudes of the neighbour pixels around the key points are calculated (See Fig (d))
- Histograms of gradient orientations are then created according to gradient magnitudes. Finally, a 128-dimensional vector is computed and assigned to related key points for further matching processes (See Fig(e))

How to achieve scale invariance

- Pyramids

- Scale Space (DOG method)

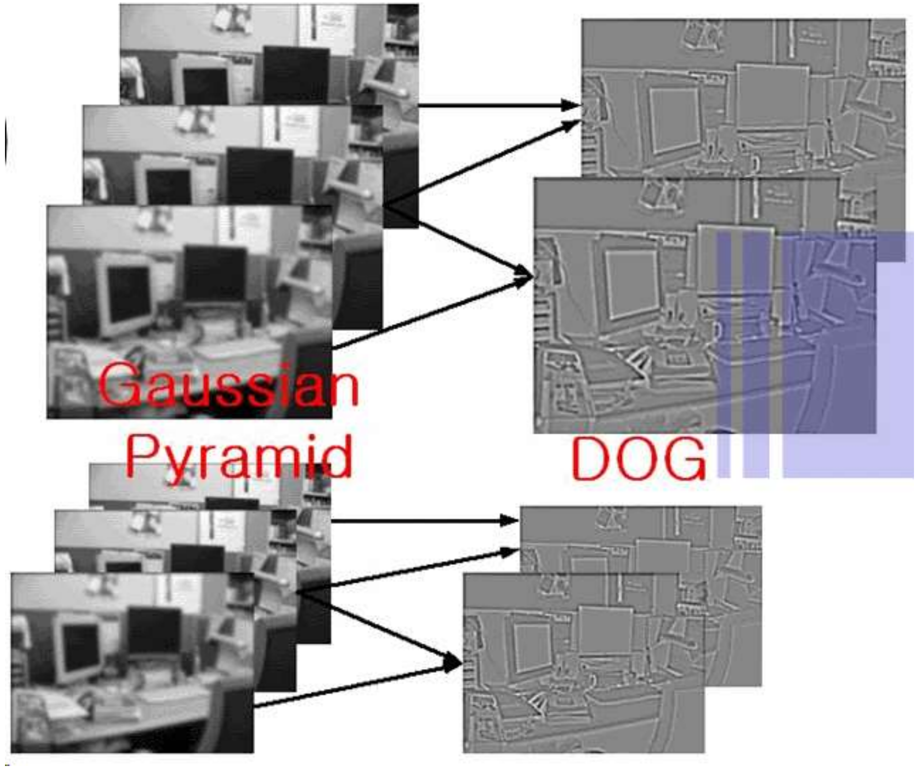
Pyramids

- Divide width and height by 2
- Take average of 4 pixels for each pixel (or Gaussian blur with different σ)
- Repeat until image is tiny
- Run filter over each size image and hope its robust

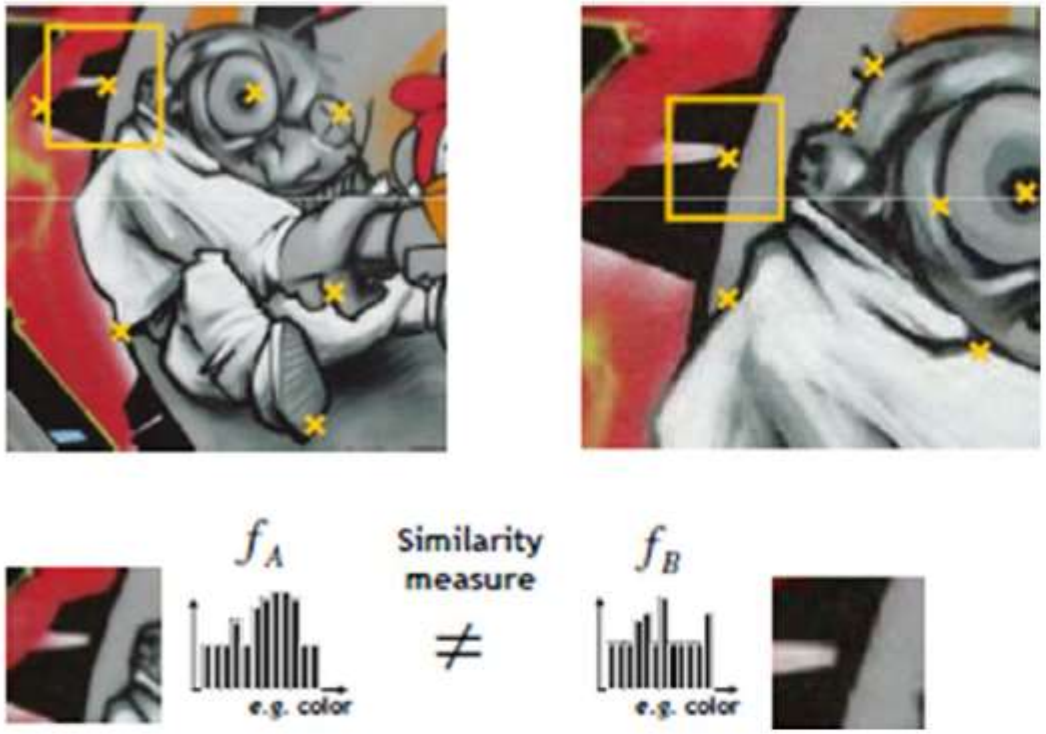
How to achieve scale invariance

- Scale Space: Difference of Gaussian (DOG)
  - Take DOG features from differences of these images-producing the gradient image at different scales.
  - If the feature is repeatedly present in between Difference of Gaussians, it is Scale Invariant and should be kept.

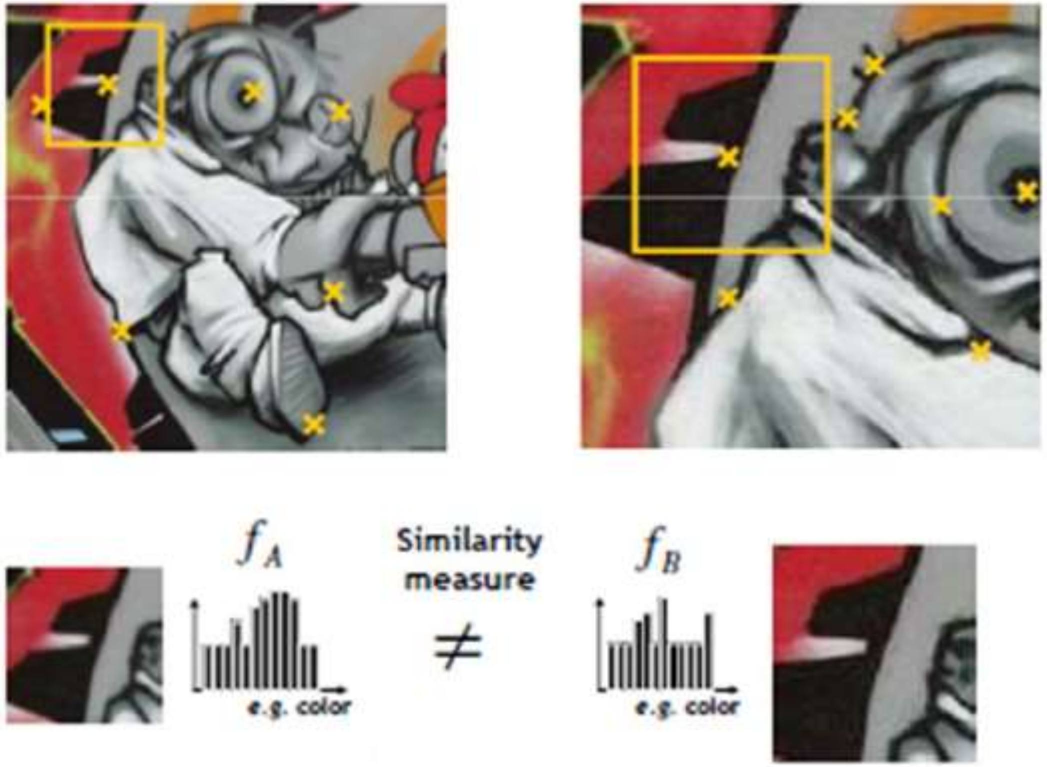
Differences Of Gaussians



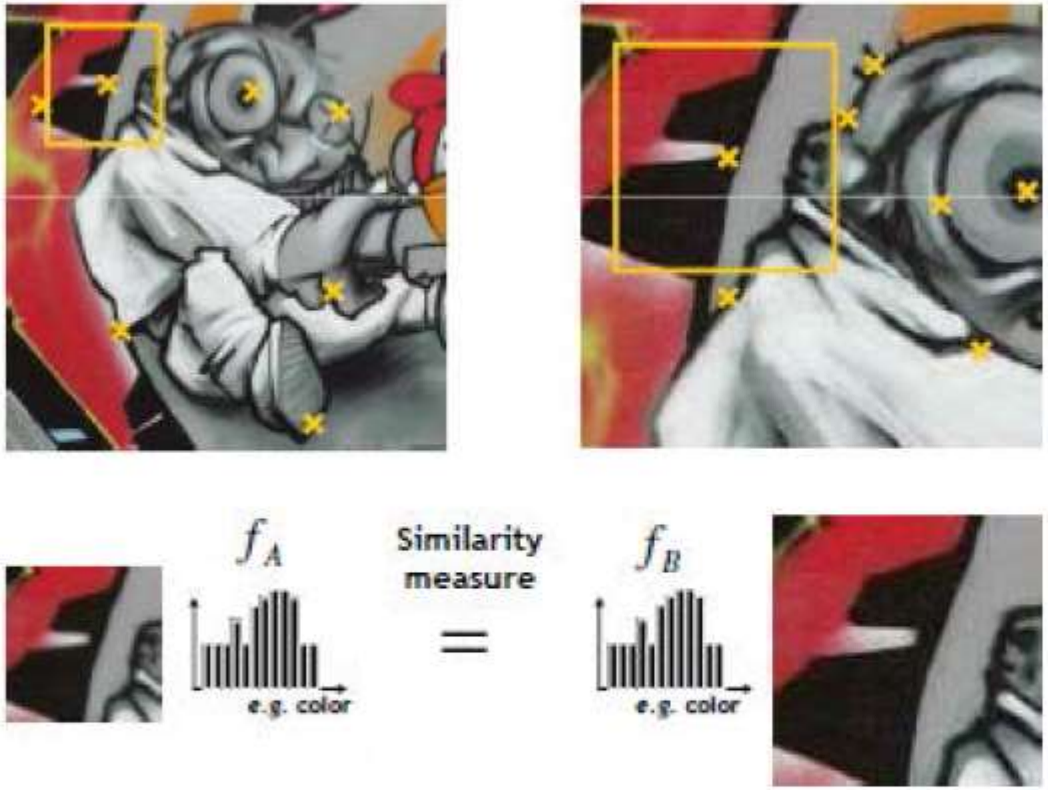
Scale Different



Exhaustive Search

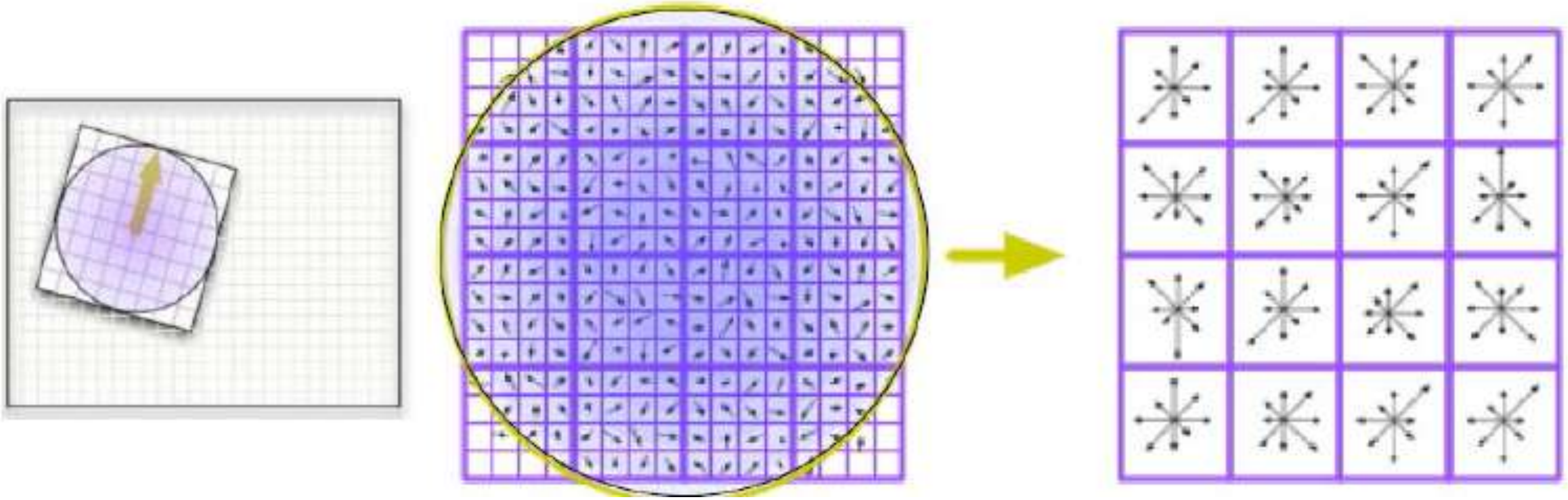


Try Different Window Sizes



SIFT Feature Calculation

- Take the region around a keypoint according to its scale
- Rotate and align with the previously calculated orientation
- 8 orientation bins calculated at 4x4 bin array
-



Illumination Issues

- SIFT is a 128 dimensional vector
- For robustness to illumination, normalize the feature to unit magnitude
- To cater for image saturations, truncate the feature to 0.2
- Renormalize to unit magnitude
- SIFT properties
  - Repeatable keypoints
  - Scale invariant
  - Rotation invariant
  - Robust to viewpoint
  - Robust to illumination changes