Sharif University of Technology
Electrical Engineering Department

Machine Learning and Vision Lab
Pre-Report 6

Amir Hossein Yari
99102507

November 26, 2023

Contents

LOG Proof

We know that: σ:

$$G(x,y;\sigma) = \frac{1}{2\pi\sigma^2}e^{-\frac{(x+y)^2}{2\sigma^2}}$$

$$\frac{\partial G}{\partial \sigma} = \sigma \nabla^2 g$$

$$\nabla^2 g(x,y;\sigma) = -\frac{1}{2\pi\sigma^4}\left(2 - \frac{x^2+y^2}{\sigma^2}\right)e^{-\frac{x^2+y^2}{2\sigma^2}}$$

Let's substitute this expression into the partial derivative:

$$\frac{\partial G}{\partial \sigma} = \sigma\left(-\frac{1}{2\pi\sigma^4}\left(2 - \frac{x^2+y^2}{\sigma^2}\right)e^{-\frac{x^2+y^2}{2\sigma^2}}\right)$$

Now, let's integrate this expression with respect to σ:

$$\text{LOGL}(\sigma) = \int \frac{\partial G}{\partial \sigma} d\sigma$$

$$= \int \sigma\left(-\frac{1}{2\pi\sigma^4}\left(2 - \frac{x^2+y^2}{\sigma^2}\right)e^{-\frac{x^2+y^2}{2\sigma^2}}\right)d\sigma$$

Let's simplify the expression by factoring out constants:

$$= -\frac{1}{2\pi}\int\left(2\sigma^{-3} - \frac{x^2+y^2}{\sigma^{-1}}\sigma^{-3}\right)e^{-\frac{x^2+y^2}{2\sigma^2}}d\sigma$$

$$= -\frac{1}{2\pi}\int\left(2\sigma^{-3} - \frac{x^2+y^2}{\sigma^2}\right)e^{-\frac{x^2+y^2}{2\sigma^2}}d\sigma$$

Now, let's make a substitution $t = \frac{x^2+y^2}{2\sigma^2}$, then $dt = -\frac{x^2+y^2}{\sigma^3}d\sigma$:

$$= \frac{1}{4\pi}\int(2t+1)e^{-t}dt$$

This integral can be evaluated to give:

$$= \frac{1}{4\pi}\left[-(2t+2)e^{-t} + \int 2e^{-t}dt\right]$$

$$= \frac{1}{4\pi}\left[-2(1+t)e^{-t} - 2e^{-t}\right]$$

Now, substitute back $t = \frac{x^2+y^2}{2\sigma^2}$:

$$= -\frac{1}{2\pi}(1 + \frac{x^2+y^2}{\sigma^2})e^{-\frac{x^2+y^2}{2\sigma^2}}$$

Now, let's evaluate the expression at $K\sigma$ and σ:

$$\text{LOGL}(K\sigma) = -\frac{1}{2\pi}\left(1 + \frac{x^2+y^2}{(K\sigma)^2}\right)e^{-\frac{x^2+y^2}{2(K\sigma)^2}}$$

$$\text{LOGL}(\sigma) = -\frac{1}{2\pi}\left(1 + \frac{x^2+y^2}{\sigma^2}\right)e^{-\frac{x^2+y^2}{2\sigma^2}}$$

Now, subtracting these two expressions:

$$\text{LOGL}(K\sigma) - \text{LOGL}(\sigma) = -\frac{1}{2\pi}\left(\frac{x^2+y^2}{(K\sigma)^2} - \frac{x^2+y^2}{\sigma^2}\right)e^{-\frac{x^2+y^2}{2\sigma^2}}$$

$$= \frac{1}{2\pi\sigma^2}\left(1 - \frac{1}{K^2}\right)e^{-\frac{x^2+y^2}{2\sigma^2}}$$

Now, multiplying both sides by $K-1$:

$$(K-1)(\text{LOGL}(K\sigma) - \text{LOGL}(\sigma)) = \frac{1}{\sigma^2}[G(x,y;K\sigma) - G(x,y;\sigma)]$$

Finally, dividing both sides by $K-1$:

$$\text{LOGL}(\sigma) \approx \frac{1}{(K-1)\sigma^2}[G(x,y;K\sigma) - G(x,y;\sigma)]$$

So, the given approximation is derived.

SIFT Method

1. **Scale-space Extrema Detection:**
- SIFT uses a scale-space representation of the image, created by applying Gaussian blurring and downsampling at different scales. This results in a series of images at multiple scales.
- At each scale, the algorithm detects key points by looking for extrema (maxima or minima) in the difference-of-Gaussian (DoG) images, which are obtained by subtracting adjacent blurred images.

2. **Keypoint Localization:**
- Once potential key points are identified, SIFT performs keypoint localization to refine their positions. It fits a 3D quadratic function to the nearby points in the DoG scale space to determine the precise location of the keypoint.
- The localization process involves considering the intensity values and their gradients. This helps in accurately localizing the keypoint even in the presence of noise.

3. **Orientation Assignment:**
- To achieve rotation invariance, SIFT computes a dominant orientation for each keypoint. It considers the gradient magnitudes and orientations in the local neighborhood of the keypoint.
- The orientation is assigned based on the histogram of gradient orientations, and the keypoint descriptor is then rotated accordingly to make it invariant to rotations.

4. **Descriptor Calculation:**
- SIFT generates a descriptor for each keypoint to represent the local image information around that point. The descriptor captures information about the gradients of intensity in the keypoint's neighborhood.
- The descriptor is computed relative to the assigned orientation, making it invariant to rotations.

K-Means

K-Means clustering is an iterative algorithm used for partitioning a dataset into K distinct, non-overlapping subsets (clusters). The objective is to minimize the variance within each cluster and maximize the variance between different clusters. The algorithm operates as follows:
1. **Initialization:** Randomly select K data points from the dataset as the initial cluster centers.
2. **Assignment:** Assign each data point to the nearest cluster center, usually based on Euclidean distance.
3. **Update:** Recalculate the cluster centers as the mean of all data points assigned to each cluster.
4. **Repeat:** Repeat the assignment and update steps until convergence, where convergence occurs when the cluster assignments no longer change significantly.
5. **Result:** The final result is a set of K clusters, each represented by its centroid.

Dependency on Initial Cluster Centers:
Yes, the choice of initial cluster centers in K-Means can significantly impact the final clustering result. Different initializations may lead to different local minima for the clustering objective function.

Improving Independence from Initial Centers:
To make K-Means less dependent on initial cluster centers, several strategies can be employed:
1. **K-Means++ Initialization:** Use a smarter initialization method, like K-Means++, which selects initial centroids in a way that encourages a more spread-out initial configuration.
2. **Hierarchical K-Means:** Employ a hierarchical version of K-Means, where clusters are formed at different scales, potentially reducing sensitivity to initializations.

K-Means++ Algorithm:
1. **Select the First Center at Random:**
- Choose the first cluster center uniformly at random from the data points.
2. **Compute Distances and Assign Probabilities:**
- For each data point, calculate its distance to the nearest existing cluster center.

- Assign a probability to each data point proportional to the square of its distance from the nearest cluster center.

3. **Select Subsequent Centers Randomly with Probability Weighting:**

- Randomly select the next cluster center from the data points, with the probability of each point being proportional to its squared distance from the nearest existing cluster center.

4. **Repeat:** - Repeat steps 2 and 3 until K cluster centers are chosen.