# Scale-Invariant Feature Transform

## Team 4: Drowning in Inefficiency


- Kannav Mehta (2019101044)
- Raj Maheshwari (2019101039)
- Triansh Sharma (2019101006)
- Aashwin Vaish (2019114014)


Github link

# Project Objectives

The main obective of the project is to implement the SIFT algorithm described in the paper by David G. Lowe [1] from scratch(without any computer-vision dependencies) and using said implementation, we tackle the problem of automated panoramic image stitching by developing an image-stiching program that uses SIFT features to create panoramas from set of images [2] as shown here.

(a) Original images and blended result

# Goals and Approach

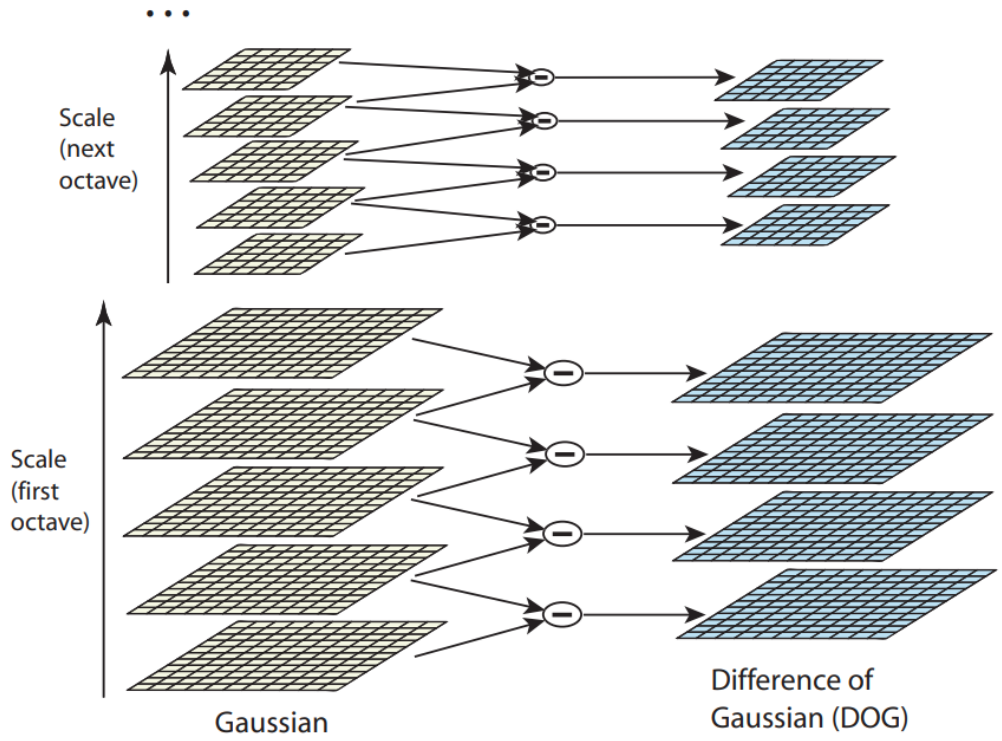
Our goal is to develop an invariant feature based approach to full automatic panoramic image stiching.The first step is to extract all the features from all the images, for this we are using SIFT[1].

## SIFT

Following are the major stages of computation used to generate the set of image features using SIFT are:

### Scale-space extrema detection

The first stage of computation searches over all scales and image locations. It is implemented efficiently by $D(x, y, \sigma)$, obtained by convolving an image with a difference of Gaussian function to identify potential interest points that are invariant to scale and orientation. It provides a close approximation to the scale-normalized Laplacian of Gaussian.




The scale space of an image is defined as a function, $L(x, y, \sigma)$, that is produced from the convolution of a variable-scale Gaussian, $G(x, y, \sigma)$, with an input image, $I(x, y)$

$$L(x, y, \sigma) = G(x, y, \sigma) * I(x, y)$$

where $*$ is the convolution operation in $x$ and $y$, and

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

$$\begin{aligned} D(x, y, \sigma) &= (G(x, y, k\sigma) - G(x, y, \sigma)) * I(x, y) \\ &= L(x, y, k\sigma) - L(x, y, \sigma) \end{aligned}$$

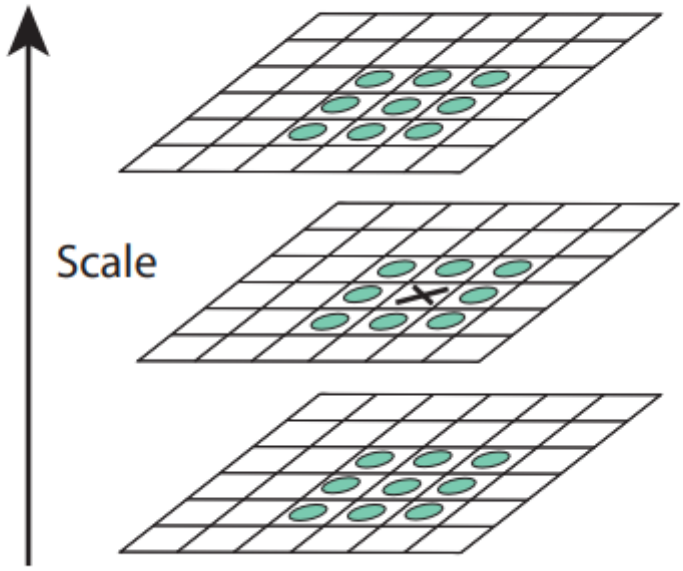
The Laplacian of Gaussian is given by

$$\sigma\nabla^2 G = \frac{\partial G}{\partial \sigma} \approx \frac{G(x, y, k\sigma) - G(x, y, \sigma)}{k\sigma - \sigma}$$

This gives us a close approximation to the scale-normalised Laplacian of Gaussian ($\sigma^2\nabla^2 G$) because the constant factor $k - 1$ does not influence extreme location.

## Keypoint localization

Once each candidate is found by comparing its pixel value with its neighbours, a detailed model is fit to determine location, scale and principal curvatures.




Keypoints are selected based on measures of their stability. The location of the extremum, $\hat{x}$, is determined by taking the derivative of taylor expansion of previously obtained function $D(x, y, \sigma)$ with respect to $x$ and setting it to zero, giving:

$$\hat{x} = -\frac{\partial^2 D}{\partial x^2}^{-1} \frac{\partial D}{\partial x}$$

where taylor expansion of $D(x, y, \sigma)$ is given by

$$D(x) = D + \frac{\partial D^T}{\partial x} x + \frac{1}{2} x^T \frac{\partial^2 D}{\partial x^2} x$$

To obtain more stable fixed points, we apply a threshold on the value of $|D(\hat{x})|$ to reject key points with low contrast. Further, we eliminate those key points whose ratio for principal curvatures fall below a certain threshold.

We define Hessain $H$ of $D$ as

$$H = \begin{bmatrix} D_{xx} & D_{xy} \\ D_{xy} & D_{yy} \end{bmatrix}$$

The eigen values $\alpha$ and $\beta$ of $H$ are proportional to the principal curvatures of $D$, hence we define the threshold by $\tau$, $(\alpha = r\beta)$

$$\tau = \frac{Tr(H)^2}{Det(H)} < \frac{(r+1)^2}{r}$$

## Orientation assignment

One or more orientations are assigned to each keypoint location based on local image gradient directions. All future operations are performed on image data that has been transformed relative to each feature's set orientation, scale, and location, thereby providing invariance to these transformations.

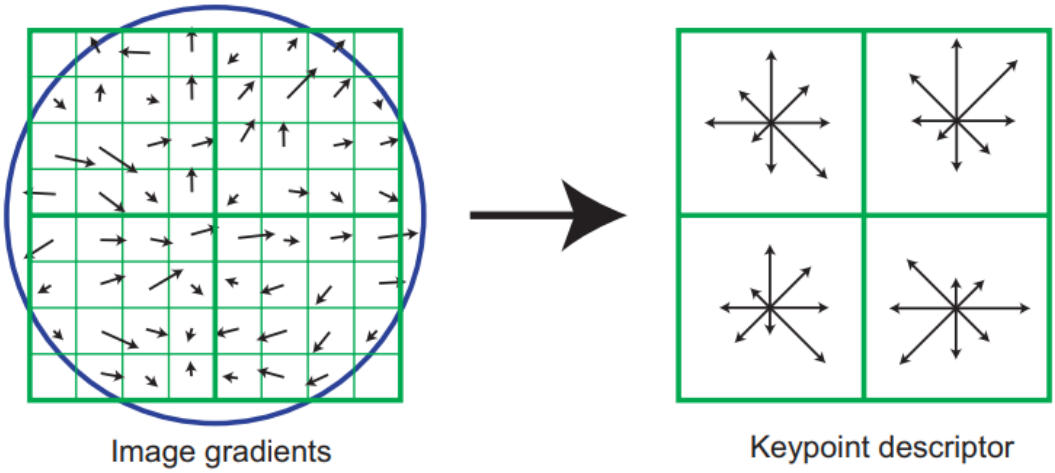
The scale of the keypoint is used to select the Gaussian smoothed image, $L$, with the closest scale, so that all computations are performed in a scale-invariant manner. For each image sample, $L(x, y)$, at this scale, the gradient magnitude, $m(x, y)$, and orientation, $\theta(x, y)$, is precomputed using pixel
differences:

$$m(x,y) = \sqrt{(L(x+1,y) - L(x-1,y))^2 + (L(x,y+1) - L(x,y-1))^2}$$

$$\theta(x,y) = tan^{-1}\left(\frac{L(x,y+1) - L(x,y-1)}{L(x+1,y) - L(x-1,y)}\right)$$

### Keypoint descriptor

The local image gradients are measured at the selected scale in the region around each keypoint. These samples are then accumulated into orientation histograms summarising the contents over 4x4 subregions, as described in the figure, with the length of each arrow corresponding to the sum of the gradient magnitudes near that direction within the region calculated using a Gaussian weighted function. This allows for significant levels of local shape distortion and change in illumination.




## Getting Matches between pair of Images using KNN

Once features have been extracted from all $n$ images (linear time), they must be matched. Since multiple images may overlap a single ray, each feature is matched to its $k$ nearest neighbours in feature space. This can be done in $O(n\ log\ n)$ time by using a *k-d tree* to find the approximate nearest neighbours.
After the above step, we will have identified images that have a large number of matches between them and connected sets of these image matches that will later become panoramas.

# Homography Estimation using RANSAC

RANSAC (random sample consensus) is a robust estimation procedure that uses a minimal set of randomly sampled correspondences to estimate image transformation parameters and finds a solution that has the best consensus with the data.  In the case of panoramas, we select sets of $r$ feature correspondences and compute the homography $\mathbf{H}$ between them using the direct linear transformation. Given the probability that a feature match is correct between a pair of matching images (the inlier probability) is $p_i$, the probability of finding the right transformation after $n$ trials are:

$$P(\mathbf{H}\ is\ correct) = 1 - (1 - p_i^r)^n$$

After a large number of trials, the probability of finding the correct homography is very high. RANSAC is essentially a sampling approach to estimating $\mathbf{H}$. For each pair of potentially matching images, we have a set of geometrically consistent feature matches ( RANSAC inliers) and many features inside the area of overlap but not consistent ( RANSAC outliers). The idea of our verification model is to compare the probabilities that this set of inliers/outliers was generated by a correct image match or by a false image match.

**The final algorithm looks as follows:**

**Algorithm: Automatic Panorama Stitching**

**Input:** $n$ unordered images

I. Extract SIFT features from all $n$ images

II. Find $k$ nearest-neighbours for each feature using a k-d tree

III. For each image:
  (i) Select $m$ candidate matching images that have the most feature matches to this image
  (ii) Find geometrically consistent feature matches using RANSAC to solve for the homography between pairs of images
  (iii) Verify image matches using a probabilistic model

IV. Find connected components of image matches

V. For each connected component:
  (i) Perform bundle adjustment to solve for the rotation $\theta_1, \theta_2, \theta_3$ and focal length $f$ of all cameras
  (ii) Render panorama using multi-band blending

**Output:** Panoramic image(s)

SIFT features are extracted from all of the images. After matching all of the features using a k-d tree, the m images with the most significant number of feature matches to a given image are checked for an image match. First RANSAC is performed to compute the homography, and then a probabilistic model is invoked to verify the image match based on the number of inliers. For each pair that have a consistent set of feature matches, connected components of image matches are detected and stitched into panoramas.

# Deliverables

Given the timeframe, we aim to deliver the following:

- A minimal-dependency algorithm implementation to classify images belonging to the same scene using distinct feature detection.
- A program to stitch several images belonging to a panorama using the implementation mentioned above and blend them with a multi-band blurring algorithm.
- Presentation and description of the project along with instructions to run the demo.

## Milestones and Timeline

| When | What |
|---|---|
| Nov 7 | Project Proposal |
| Nov 8 -- Nov 16 | SIFT implementation from scratch |
| Nov 16 -- Nov 20 | Mid term evaluation |
| Nov 20 -- Dec 1 | Implementing image stitching algorithm |
| Dec 1 -- Dec 4 | Final project presentation |

## Work Distribution

Work will be distributed equally, and all members will be free to contribute to any part. Everyone's ideas will be respected.

## References


1. *Brown, M., Lowe., D.G. Recognising panoramas. In proceedings of the 9th International Conference on Computer Vision (ICCV03), volume 2, pages 1218–1225, Nice, October 2003.*
2. *Lowe, D.G. Distinctive Image Features from Scale-Invariant Keypoints. International Journal of Computer Vision 60, 91–110 (2004).*
3. *Brown, M., Lowe, D.G. Automatic Panoramic Image Stitching using Invariant Features. Int J Comput Vision 74, 59–73 (2007).*