iRISE

I'm Reverse Image Search Engine

iRISE - Introduction

- What is Reverse Image Search??
  - Reverse image search is different kind of search where one does not provide keywords and instead provides an image and the search engine searches for images of similar kind.
  - It uses what is called as Content-based image retrieval (CBIR), also known as query by image content (QBIC) and content-based visual information retrieval (CBVIR) .CBIR is the application of computer vision techniques to the image retrieval problem, that is, the problem of searching for digital images in large databases.

iRISE - Introduction

- Why Reverse Image Search ??
  - Not every piece of information can be accurately translated into text
  - Increasing amounts of data
  - Image Search basic to Unique Identification
  - Uses image-identification technology rather than keywords, metadata, etc;

iRISE- How does it work?

- iRISE works on the basic principle of matching of features of an Image.
- Feature Extraction – Accomplished using Scale Invariant Feature Transform
- Feature Matching - ”Fuzzy Matching” of Image Features

Scale Invariant Feature Transform

- And its not just scale Invariant

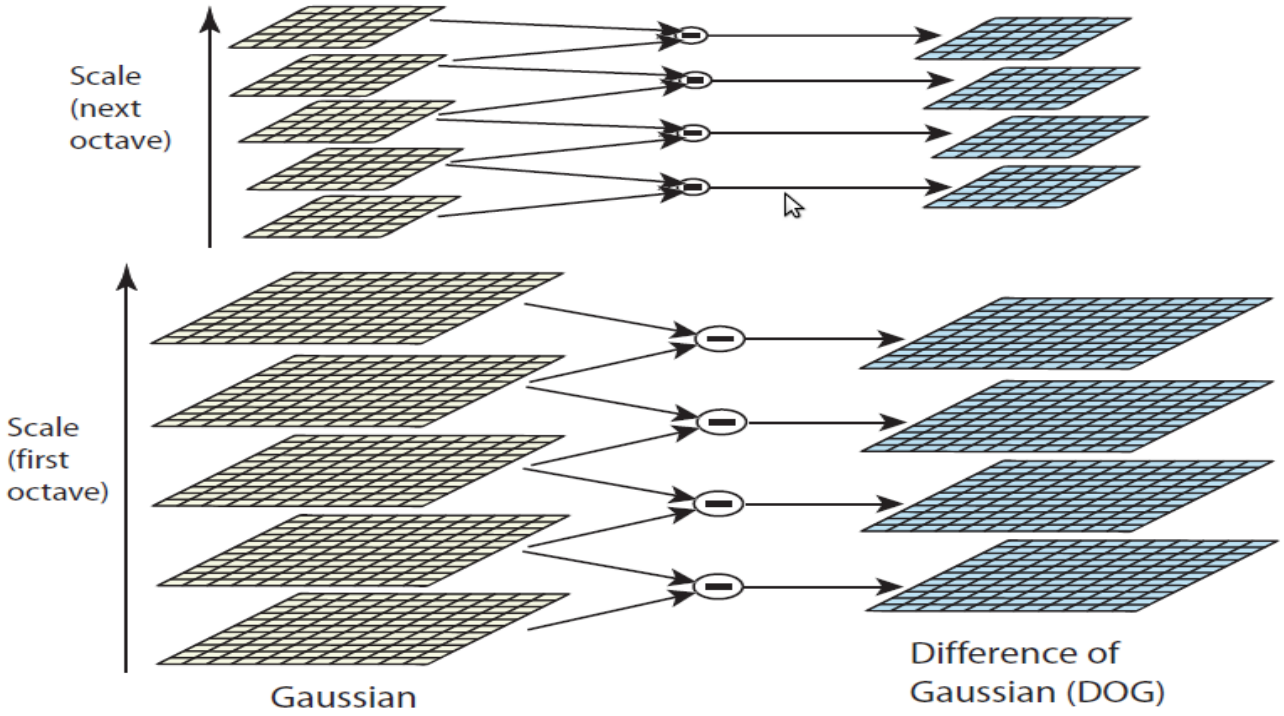
iRISE- Extracting Features

- Scale-invariant feature transform (or SIFT) is an algorithm in computer vision to detect and describe local features in images. The algorithm was published by David Lowe in 1999.
- The following are the major stages of computation:
  - Scale-space extrema detection
  - Keypoint localization
  - Orientation assignment
  - Keypoint descriptor

iRISE- Extracting Features

Scale-Space Extrema Detection

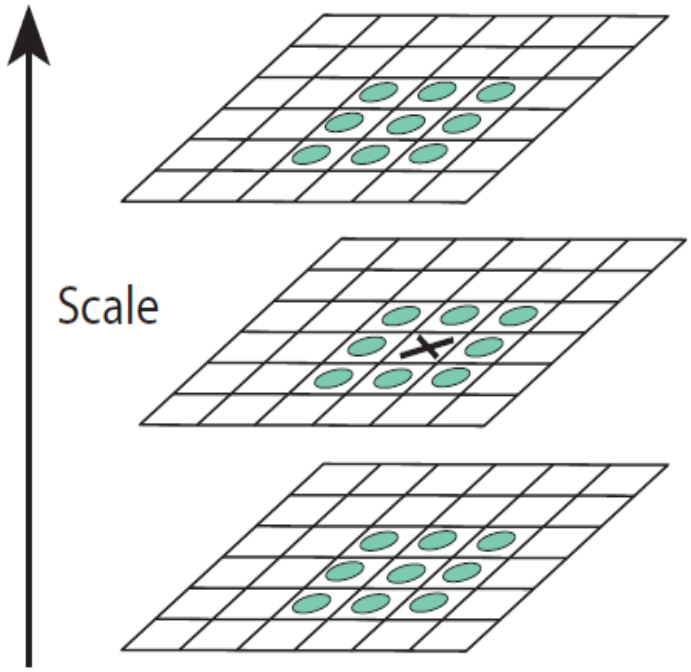
- The first stage of computation searches over all scales and image locations. It is implemented efficiently by using a difference-of-Gaussian function to identify potential interest points that are invariant to scale and orientation.



iRISE- Extracting Features

KeyPoint Localization

At each candidate location, a detailed model is fit to determine location and scale. Keypoints are selected based on measures of their stability.



iRISE- Extracting Features

Keypoint descriptor

The local image gradients are measured at the selected scale in the region around each keypoint. These are transformed into a representation that allows for significant levels of local shape distortion and change in illumination.

iRISE- Extracting Features

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 the assigned orientation, scale, and location for each feature, thereby providing invariance to these transformations.

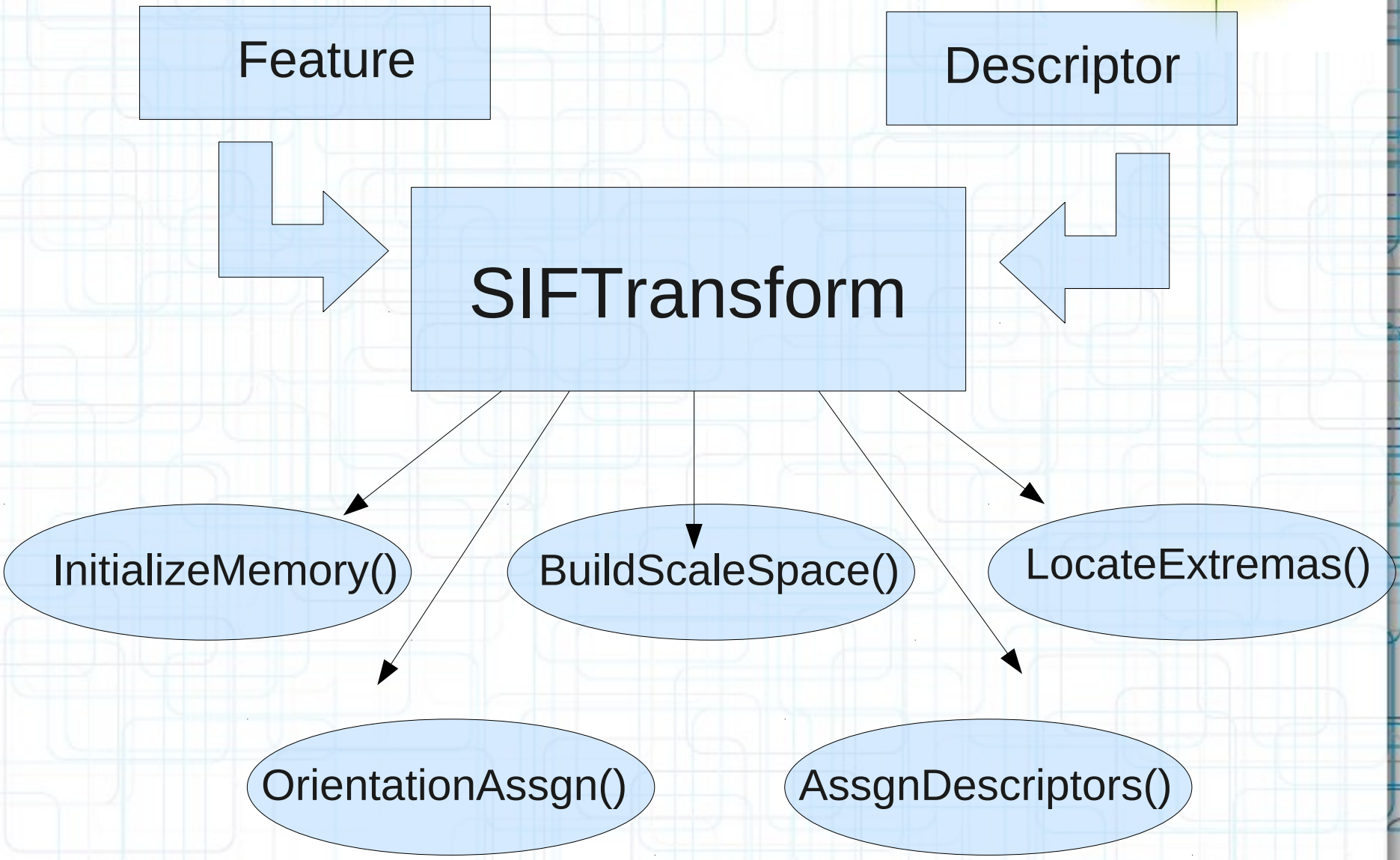
iRISE
Structure and Implementation

iRISE
Structure and Implementation

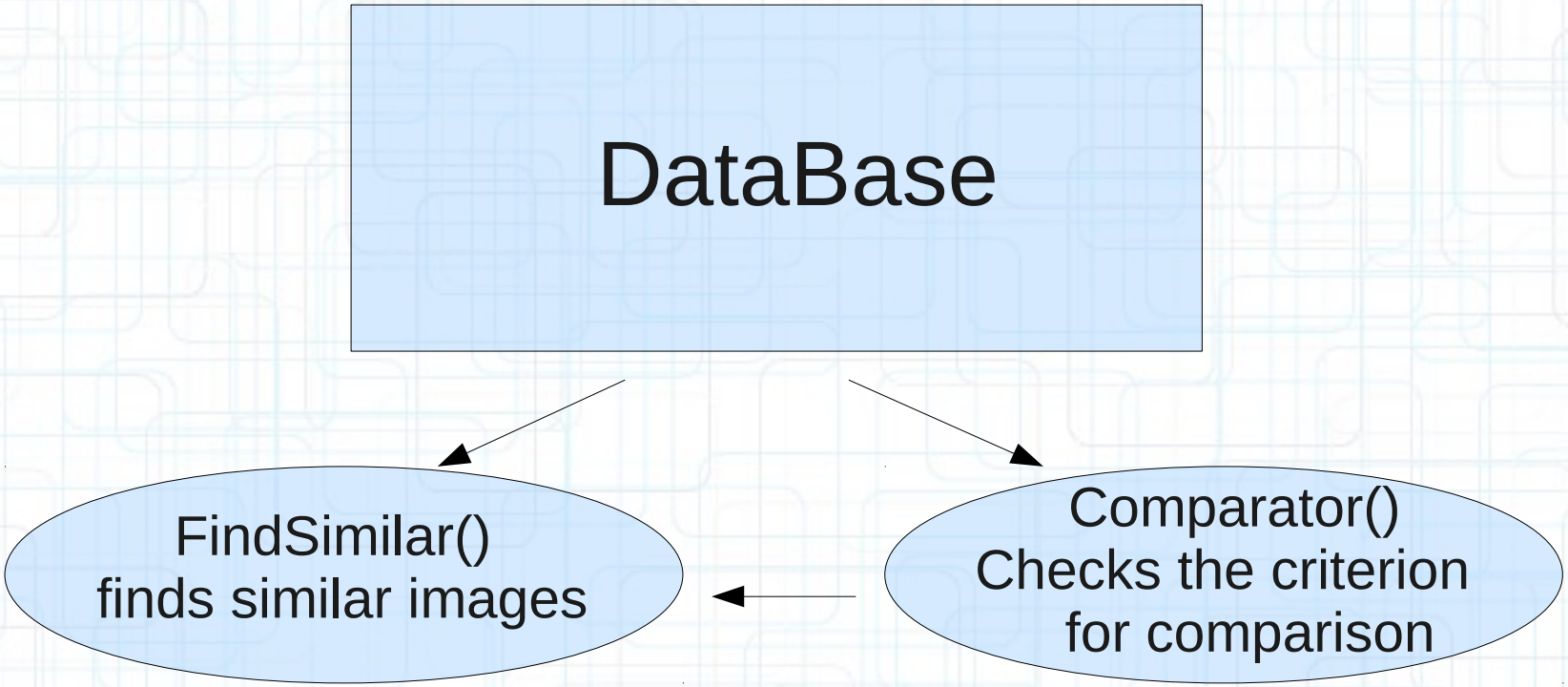
The project was trisected into 3 modules:

- Implementation of SIFT
  - Used OpenCV library for the same
- Matching in DataBase
  - Used ”fuzzy Matching”
  - Calculations done using python
- Graphics-User Interface
  - Use of FLTK

Feature Detection Implementation



Feature Matching Implementaion



Algorithmic Aspects

- **SIFT Implementation**-The toughest job (and the most time-consuming) was to understand, comprehend, and further code the SIFT (Scale Invariant Feature Transform) Algorithm.
- **Distances between the descriptors-** The Descriptor vectors for the Features had too small component values. Hence, finding out the distances between each pair of descriptor was difficult to be implemented in C++ because of the precisional losses.
- **Matching Algorithm-** Based on ”fuzzy-matching” of the Euclidian Distances between 2 the descriptors of the image

Further possible Modifications

- If time permitted, we could improvise the matching algorithm for time efficiency by stage-by-stage rejection of various images based upon primary comparison of hashed descriptor-distances
- We could improvise upon the Graphics-User interface

Work Distribution

Umang Mathur

- Read and understood the SIFT algorithm
- SIFT Implementation in OpenCv
- Implemented the Matching Algorithm
- DataBase Handling (partly done)
- Slides and reports
- Amounts to 50%

Work Distribution

Piyush Kumar

- SIFT Implementation in OpenCv (partly done)
- Designed the Algorithm for matching descriptors
- Descriptor distances in Python
- DataBase Handling (partly done)
- GUI in FLTK
- Amounts to 50%

Thank You

Hope You Enjoyed the working of iRISE