CHAPTER 1
INTRODUCTION

Image annotation in today's world is considered to be one of the most diverse and the most challenging problem in today's computer vision informatics. Image annotation is the labeling of individual image and differentiation of various image based on the category that they belong to if any. There are primarily two types of image annotations out there, one is manual and the other is automatic image annotation. Through manual annotation has been done for a very long time but the significant problem with manual annotation is the fact that it is very resource consuming and the need for human intervention is very significant. Automatic image annotation solves this problem with the use of Machine Learning algorithms and feature detectors but with the downside of being it is not as accurate as the manual annotation. One of the reasons for this is an almost complete lack of human intervention. Automatic Annotation, on the other hand, compensates for this by being 100 times faster and can handle an exponential order of image datasets as compared to manual annotation. The biggest applications of Automatic Image annotation are in the field of image search and retrieval. Some of the biggest search engines today like Google Image Search and DuckDuckGo rely heavily on the textual information that accompanies an image while performing an image. These image search engines, therefore, do not concentrate on the content in the images rather they focus on the points of interest in a particular image. Henceforth, looking at all these facts, using manual annotation will give out spurious results that cannot be trusted.

A well-programmed system that can accurately annotate image datasets can be proved to be very helpful in industrial application. Though the main aim of any image annotation system is this. Hence a mechanized framework that can precisely propose comment watchwords for pictures in the wake of breaking down its substance can really turn out to be extremely valuable. In spite of the fact that this is a definitive objective of any picture comment framework, functional and computational issues put a requirement on the number of affiliations that can be learned amongst watchwords and the sections in an image. The issue of programmed picture comment is firmly identified with that of Content-Based Image Recovery (CBIR) [3]. Since the mid-1990s, various methodologies, both from the scholarly world and the business, have been proposed to file pictures utilizing numerical highlights consequently removed from the pictures.

The shared opinion for CBIR [3] frameworks is to extricate a mark for each picture in view of its pixel esteems and to characterize a run for looking at images. The segments of the mark are called highlights. One preferred standpoint of a mark over the first pixel esteems is the critical pressure of picture portrayal. In any case, a more imperative explanation behind utilizing the mark is to pick up on an enhanced relationship between picture portrayal and semantics. All things considered, the fundamental errand of outlining a mark is to overcome any issues between picture semantics and the pixel portrayal, that is, to make a superior connection with picture semantics.

For the completion of this project, MATLAB is used as it has plenty of libraries available to support image processing along with machine learning algorithms [4]. Also, regarding the implementation part of the project, an image database of 4000+ image is chosen. Caltech database is the perfect choice for this project as the database consists of six categories namely airplanes_side, background, cars, cars_bg, faces, and motorbikes_side. The database is then fed into a bag-of-features. The main aim of bag-of-features is to detect and extract the features from all the test images in the database. For the detection and extraction of features in the sets of images, Speeded Up Robust Features (SURF) [2] is used. SURF was chosen over many of the feature detectors as it was computationally the fastest and the most reliable when it came to deciding on this particular project. Bag-of-features [2] computes the necessary features required for classification which are then used upon K-Means Clustering algorithm[2]. The aim of using a clustering algorithm is to segregate the features of a particular image or a particular category to help the use of machine learning algorithm in order to differentiate successfully between various categories in the end. After the clusters are formed using the K-Means algorithm, the resultant is used as an input to the Support Vector Machine (SVM). A Support Vector Machine (SVM) [2] is machine learning algorithm that is primarily used for classification of the image that primarily uses planes of the intersection to separate clusters and remember the clusters separated.

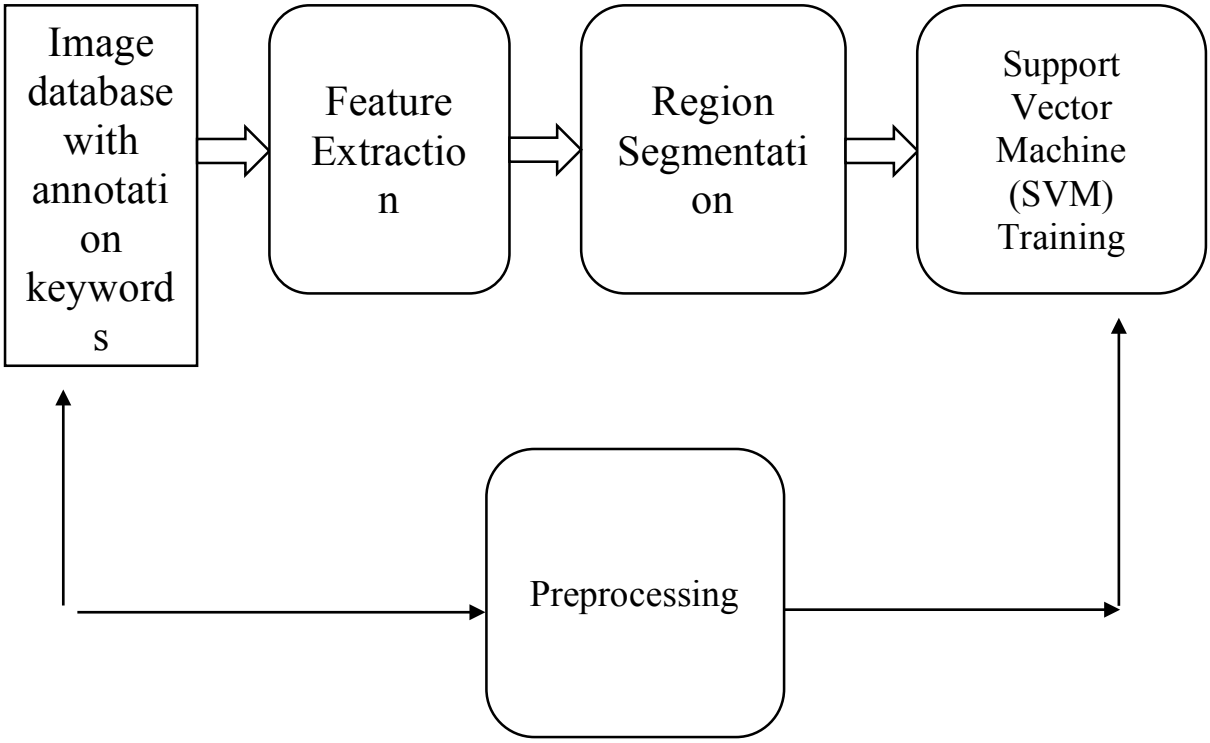

Fig 1: The Training Process

After the SVM computations are finished, a test window of 5 random images is used to display the efficiency of the project. Furthermore, the accuracy and recall of the project are calculated so as to provide some quantitative measures.

1.1 PRE-PROCESSING

The need for a database is first most step to be used in the project. Hence, getting this step done right is crucial. The database chosen for this particular project is provided by Caltech. The database consists of 4000+ images. The database provided is unorganized hence required the urgent need to perform preprocessing on the database typically based on the categories most common. The need for this resulted in the categorization of the whole database into six categories namely airplanes_side, background, cars, cars_bg, faces, and motorbikes_side. Although many more categories could have been possible that would have only increased the

strain on the support vector machine which would have diluted the accuracy and efficiency of the algorithm altogether.

After the categorization of the whole database is done, now comes the need to decide the ratio of the training set versus the validation set. The Training Set is that image dataset upon which the support vector machine is trained upon. It is one of the most fundamental parts of the training process as the rest of the project entirely depends upon this. The validation set is that set of images upon which the project will be evaluated and hence will yield the accuracy and efficiency. The ratio decided here was 50:50. Hence, almost 50% of the images are used for the training dataset and the rest 50% is used for the validation dataset.

Moving forward was the idea to randomize the entire dataset irrespective of the nature of the set, i.e., the validation and the test datasets. This was done in order to minimize any order of overlap or to eliminate the possibility of repeated images in a single order.

CHAPTER 2
MATLAB 2017b

MATLAB (matrix lab) is really a multi-paradigm numerical computing surroundings. A programming language produced by MathWorks, MATLAB allows matrix manipulations, plotting of data and functions, execution of calculations, the production of user interfaces, along with interfacing with programs written in different languages, including C, C++, C#, Java, Fortran, and Python.

Even though MATLAB is designed chiefly for numerical computing, an optional tool box employs the MuPAD search motor, allowing usage of symbolic calculating abilities. An extra package, Simulink, includes graphical multi-domain simulation and also model-based design for embedded and dynamic systems.

2.1 HISTORY

Cleve Moler, the administrator of the software engineering division at the University of New Mexico, began creating MATLAB in the late 1970s. He built it to present his students usage of LINPACK and EISPACK with no one being forced to know Fortran. It soon spread to different universities and also found that a strong crowd over the applied science community. Jack Little, an engineer, who was confronted with it within and called. They rewrote MATLAB in C and based MathWorks from 1984 to carry on its own development. These re-written libraries were understood as JACKPAC. Back in 2000, MATLAB was commissioned to make use of a broader pair of libraries for chemical manipulation, LAPACK.

MATLAB was initially adopted by professionals and researchers in control technology, with little specialization, but quickly spread to numerous different domain names. It's currently also utilized in instruction, particularly, the instruction of linear algebra, numerical analysis, and so is widely used amongst scientists involved with graphic processing.

2.2 INTERFACE

The MATLAB desktop by default contains three panels:

- The current folder
- The command window
- Workspace

The current folder window is used to access the files associated with the folder or the directory the programmer is working in. The command window is used to enter commands in the command line interface that is indicated by the prompt ">>". Users can write single or multiple commands depending upon the needs although to create a new MATLAB script (.m) is also available under "New Script". Finally, the window of Workspace is used to identify and organize the data and the user creates or imports from the .m files used.

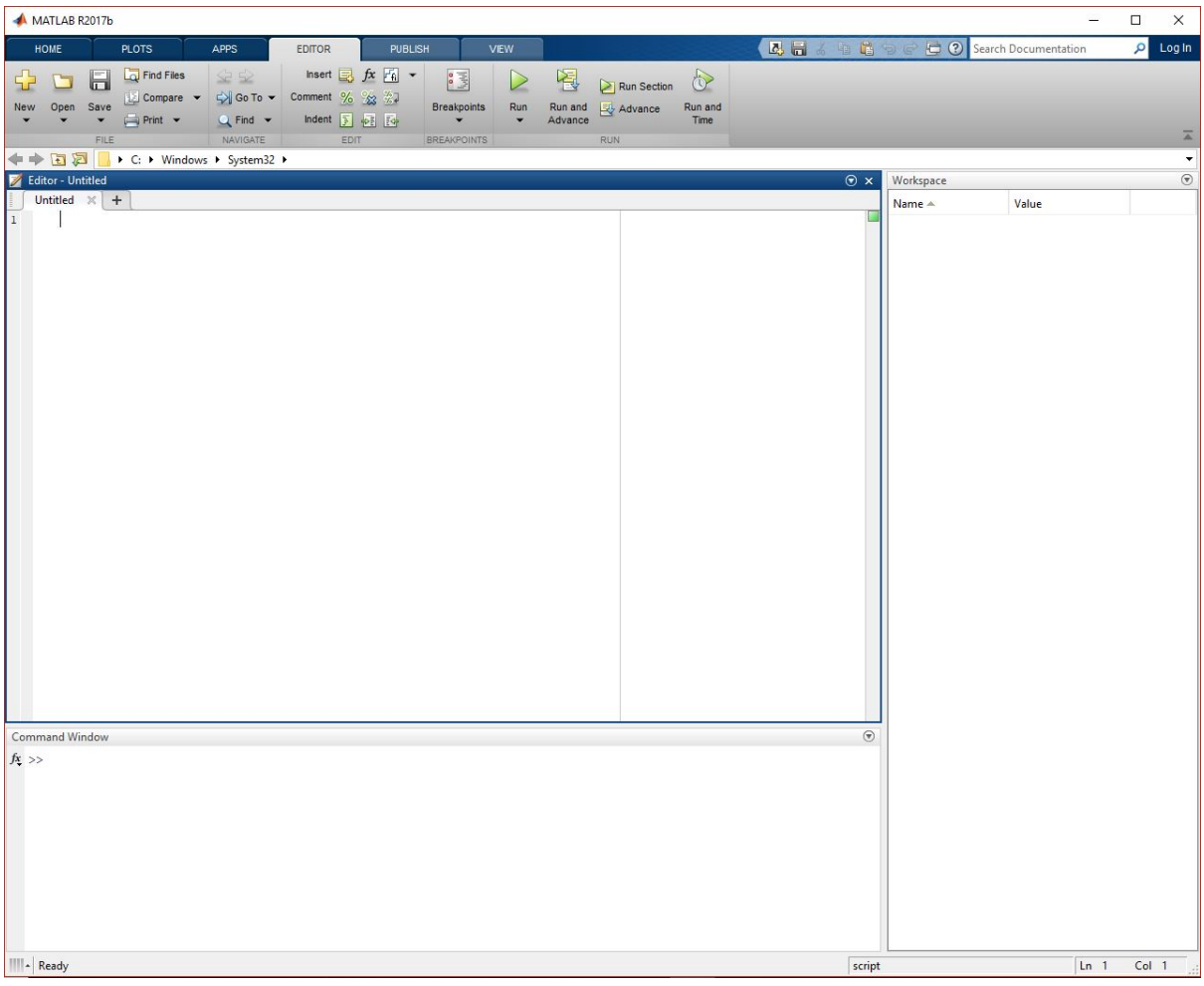

Fig 2.1 MATLAB GUI

2.3 SYNTAX

The MATLAB program is constructed round the MATLAB scripting language. Common use of this MATLAB application entails utilizing the Control Window as an interactive mathematical casing or implementing text documents comprising of the MATLAB code.

2.3.1 Variables

MATLAB is a typed programming language since forms are implicitly converted. It's an inferred typed language since factors can be delegated without announcing their kind, except when they should be treated as symbolic items, after which their kind may alter. Values can come from constants, from computation involving values of different factors, or by the outcome of a purpose. As an instance:

```
>> x = 17
x =
 17

>> x = 'hat'
x =
hat

>> y = x + 0
y =
       104        97       116

>> x = [3*4, pi/2]
x =
   12.0000    1.5708

>> y = 3*sin(x)
y =
   -1.6097    3.0000
```

Fig 2..2 MATLAB Syntax for Variables

2.3.2 Vectors and Matrices

A very simple array is defined with the colon syntax: initial: increment: terminator.

```
>> array = 1:2:9
array=
 1 3 5 7 9
```

Fig 2.3 MATLAB Syntax for Vector

This defines a variable named array or can assign a new value to a present variable with the name array, which is an array composed of the values 1, 3, 7, 5, and 9. In other words, the array begins at 1, the first value, increases with each step from the previous value by two, the increment value, and stops after it reaches 9, the terminator value or to avoid exceeding the terminator value.

```
>> A = [16 3 2 13; 5 10 11 8; 9 6 7 12; 4 15 14 1]
A =
 16  3  2 13
  5 10 11  8
  9  6  7 12
  4 15 14  1

>> A(2,3)
ans =
 11
```

Fig 2.4 MATLAB Syntax for Matrices

2.3.3 Structures

MATLAB has architectural data types. Since all variables in MATLAB are arrays, a decent title is "structure variety", where every element of this array has the exact same domain names. Moreover, MATLAB supports dynamic domain names (area look-ups by name, subject manipulations, etc.). Regrettably, MATLAB JIT doesn't encourage MATLAB structures, so only a simple bundling of different variables to a construction will probably come at a price.

2.3.4 Functions

While making a MATLAB function work, the name of the record should coordinate the name of the principal work in the document. Legitimate capacity names start with an alphabetic character and can contain letters, numbers, or underscores. Capacities are a frequently cased delicately.

2.4 INTERFACING WITH OTHER LANGUAGES

MATLAB can predict functions and subroutines written in the programming language Fortran. A wrapper function is done by allowing MATLAB data types to be returned and passed. Since 2014, a two-way integration with Python has been inserted. Libraries written in Perl, Java, Active X or even .NET could be directly predicted from MATLAB, and also lots of MATLAB libraries (as an instance XML or SQL service) are used as wrappers around Java or Active X libraries. Calling MATLAB from Java is much more complex, but can be done with some MATLAB toolbox That's sold individually by MathWorks, or even with the undocumented mechanism Named JMI, Java-to-MATLAB Interface

CHAPTER 3
FEATURE DETECTION

As clarified before, in computer vision and picture handling, a feature is a snippet of data which is important for fathoming the computational assignment identified with a specific application. This is an indistinguishable sense to highlight in machine learning and example acknowledgment by and large, however, picture preparing has an extremely advanced accumulation of highlights. Highlights might be particular structures in the picture, for example, focuses, edges or protests. Highlights may likewise be the aftereffect of a general neighborhood activity or highlight recognition connected to the picture. In computer vision and image processing, highlight identification incorporates strategies for figuring deliberations of image data and settling on nearby choices at each image point whether there is an image highlight of a given kind by then or not. The subsequent highlights will be subsets of the image space, frequently as segregated focuses, constant bends or associated areas [1].

Different cases of highlights are identified with movement in picture successions, to shapes characterized as far as bends or limits between various picture districts, or to properties of such a region. They include the idea that is exceptionally broad and the selection of highlights in a specific computer vision framework might be profoundly reliant on the particular issue within reach.

In this section, I have separated the diverse element locators like SURF, MSER, FAST, and Harris and closed the best one that can be utilized for this specific task.

3.1 DEFINITION OF A FEATURE

There is no widespread or correct meaning of what constitutes as a feature, and the correct definition regularly relies upon the issue or the kind of utilization. Given that, a component is characterized as a "fascinating" for some portion of a picture, and highlights are utilized as a beginning stage for some computer vision calculations. Since highlights are utilized as the beginning stage and principle natives for resulting calculations, the general calculation will frequently just be in the same class as its element locator. Thus, the attractive property for a

component indicator is repeatability; regardless of whether a similar element will be identified in at least two unique pictures of a similar scene.

Highlight recognition is a low-level picture handling activity. That is, it is generally executed as the principal activity on a picture and looks at each pixel to check whether there is an element exhibiting at that pixel. On the off chance that this is a piece of a bigger calculation, at that point, the calculation will normally just look at the picture in the area of the highlights. As an inherent pre-imperative to include identification, the information picture is typically smoothed by a Gaussian portion [2] in a scale-space portrayal and one or a few element pictures are processed, frequently communicated as far as nearby picture subsidiaries activities.

3.2 TYPES OF FEATURES

3.2.1 Edges

Edges are focuses where there is a limit (or an edge) between two picture districts. When all is said in done, an edge can be of a relatively self-assertive shape and may incorporate intersections. By and by, edges are normally characterized as sets of focuses in the picture which have a solid gradient greatness. Besides, some regular calculations will then chain high gradient directs together to frame a total portrayal of an edge. These calculations, as a rule, put a few imperatives on the properties of an edge, for example, shape, smoothness, and gradient value.

3.2.2 Corners/ Interest Points

The terms corners and interest points are utilized to some degree conversely and allude to point-like highlights in an image, which have a neighborhood two-dimensional structure. The name "Corner" emerged since early calculations initially performed edge detection, and after that examined the edges to discover quick alters in the course (corners). These calculations were then grown with the goal that unequivocal edge detection was never again required, for example by searching for elevated amounts of ebb and flow in the image inclination. It was then seen that the alleged corners were additionally being recognized on parts of the image which were not corners in the conventional sense (for example a little brilliant spot on a dull foundation might be identified). These points are every now and again known as interest points, yet the expression "corner" is utilized by convention.

3.2.3 Blobs / Regions of Interest Points

Blobs give an integral depiction of image structures as far as locales, instead of corners that are more point-like. All things considered, blob descriptors may regularly contain a favored point (a neighborhood greatest of an administrator reaction or a focal point of gravity) which implies that numerous blob detectors may likewise be viewed as interest point administrators. Blob detectors can identify zones in an image which are too smooth to be in any way identified by a corner identifier.

Think about contracting an image and after that performing corner detection. The locator will react to points which are sharp in the contracted image, however, might be smooth in the first image. It is now that the distinction between a corner finder and a blob locator turns out to be to some degree dubious. To a substantial degree, this qualification can be cured by including a suitable thought of scale. All things considered, because of their reaction properties to various kinds of image structures at various scales, the LoG, and DoH blob detectors are additionally said in the article on corner detection.

3.2.4 Ridges

For stretched articles, the thought of ridges is a characteristic device. An edge descriptor processed from a dim level image can be viewed as a speculation of an average pivot. From a functional viewpoint, an edge can be thought of as a one-dimensional bend that speaks to a pivot of symmetry, and moreover, has a quality of neighborhood edge width related with each edge point. Lamentably, in any case, it is algorithmically harder to extricate edge highlights from general classes of dim level images than edge-, corner- or blob highlights. By and by, edge descriptors are habitually utilized for street extraction in flying images and for separating veins in restorative images.

3.3 SURF FEATURE DETECTOR

In computer vision, Speeded Up Robust Features (SURF) is a licensed nearby feature detector and descriptor. It can be utilized for undertakings, for example, question acknowledgment, image enlistment, order or 3D recreation. It is incompletely enlivened by the Scale-Invariant Feature Transform (SIFT) descriptor. The standard rendition of SURF [2] is a few times speedier than SIFT and guaranteed by its creators to be more robust against various image transformations than SIFT.

To detect interest points, SURF utilizes an integer estimate of the determinant of Hessian blob detector, which can be figured with 3 integer tasks utilizing a precomputed fundamental image. Its feature descriptor depends on the whole of the Haar wavelet reaction around the purpose of interest. These can likewise be registered with the guide of the indispensable image. SURF descriptors have been utilized to find and perceive protests, individuals or appearances, to reproduce 3D scenes, to track questions and to extricate points of interest.

The SURF calculation depends on indistinguishable standards and ventures from SIFT; however, points of interest in each progression are extraordinary. The calculation has three fundamental parts: interest point detection, local neighborhood depiction and coordinating.

Detection

SURF utilizes square-shaped filters as an estimation of Gaussian smoothing. The SIFT approach utilizes fell filters to detect scale-invariant trademark points, where the Difference of Gaussians (DoG) is ascertained on rescaled images logically. Filtering the image with a square is considerably speedier if the integral image is utilized:

$$S(x,y) = \sum_{i=0}^{x}\sum_{j=0}^{y} I(i,j)$$

The whole of the first image inside a rectangle can be assessed rapidly utilizing the integral image, requiring assessments at the rectangle's four corners.

SURF utilizes a blob detector in light of the Hessian framework to discover points of interest. The determinant of the Hessian framework is utilized as a measure of local change around the point and points are picked where this determinant is maximal. Rather than the Hessian-Laplacian detector by Mikolajczyk and Schmid, SURF additionally utilizes the determinant of the Hessian for choosing the scale, as is likewise done by Lindeberg. Given a point p= (x, y) in an image I, the Hessian Framework H (p, σ) at any point p and a scale σ, is:

$$H(\rho,\sigma) = \begin{pmatrix} L_{xx}(\rho,\sigma) & L_{xy}(\rho,\sigma) \\ L_{yx}(\rho,\sigma) & L_{yy}(\rho,\sigma) \end{pmatrix}$$

where $L_{xx}(\rho,\sigma)$ and so forth is the convolution of the second-arrange subsidiary of Gaussian with the image *I (x, y)* at the point *x*.

The case channel of size 9×9 is a guess of a Gaussian with σ=1.2 and speaks to the most reduced level (most elevated spatial determination) for blob-reaction maps.

Scale-Space Portrayal and Area of Point of Interest

Interest points can be found at various scales, halfway on the grounds that the look for correspondences often requires correlation images where they are seen at various scales. In other feature detection calculations, the scale space is typically acknowledged as an image pyramid. Images are over and over smoothed with a Gaussian channel, at that point they are subsampled to get the following larger amount of the pyramid. Subsequently, a few stories or stairs with different measures of the covers are computed:

$$\sigma_{approx} = current\ filter\ size \times \left(\frac{base\ filter\ scale}{base\ filter\ size}\right)$$

The scale space is isolated into various octaves, where an octave alludes to a progression of reaction maps covering a multiplying of scale. In SURF, the most reduced level of the scale space is acquired from the yield of the 9×9 filters.

Subsequently, not at all like past techniques, scale spaces in SURF are executed by applying box filters of various sizes. As needs are, the scale space is broken down by up-scaling the channel measure instead of iteratively lessening the image estimate. The yield of the over 9×9 channel is considered as the underlying scale layer at scale s =1.2 (relating to Gaussian subordinates with σ = 1.2). The accompanying layers are gotten by separating the image with progressively greater veils, considering the discrete idea of integral images and the particular channel structure. This outcome in filters of size 9×9, 15×15, 21×21, 27×27... Non-most extreme suppression in a 3×3×3 neighborhood is connected to localize interest points in the image and over scales. The maxima of the determinant of the Hessian lattice are then inserted in scale and image space with the technique proposed by Brown, et al. Scale-space introduction is particularly critical for this situation, as the difference in scale between the main layers of each octave is moderately huge.

Descriptor

The objective of a descriptor is to give a special and robust portrayal of an image feature, e.g., by depicting the forced dispersion of the pixels inside the neighborhood of the purpose of interest. Most descriptors are in this way processed in a local way, subsequently, a depiction is acquired for each purpose of interest distinguished beforehand.

The dimensionality of the descriptor directly affects the two its computational intricacy and point-coordinating robustness/precision. A short descriptor [1] might be more robust against appearance varieties, however, may not offer adequate separation and hence give excessively numerous false positives. The initial step comprises of settling a reproducible introduction in light of data from a roundabout area around the interest point. At that point, we develop a square locale adjusted to the chose introduction and concentrate the SURF descriptor from it.

Introduction task

Keeping in mind the end goal to accomplish rotational invariance, the introduction of the purpose of interest should be found. The Haar wavelet reactions in both x-and y-headings inside around neighborhood of sweep *6s* around the purpose of interest are processed, where *s* is the scale at which the purpose of interest was detected. The acquired reactions are weighted by a Gaussian capacity focused at the purpose of interest, at that point plotted as points in a two-dimensional space, with the level reaction in the abscissa and the vertical reaction in the

ordinate. The overwhelming introduction is evaluated by computing the aggregate of all reactions inside a sliding introduction window of size *π/3*. The flat and vertical reactions inside the window are summed. The two summed reactions at that point yield a local introduction vector. The longest such vector general characterizes the introduction of the purpose of interest. The span of the sliding window is a parameter that must be picked precisely to accomplish the coveted harmony amongst robustness and rakish determination.

Descriptor sum of Haar Wavelet and Matching

To depict the area around the point, a square locale is removed, focused on the interest point and situated along the introduction as chose above. The extent of this window is the 20s. The interest locale is part of little 4x4 square sub-districts, and for everyone, the Haar wavelet reactions are separated at 5x5 consistently spaced example points. The reactions are weighted with a Gaussian (to offer more robustness for disfigurements, commotion, and interpretation).

Finally, by looking at the descriptors got from various images, coordinating sets can be found.

3.4 MSER FEATURE DETECTOR

Maximally Stable Extremely Regions (MSER) [7] are utilized as a method of blob detection in images. This procedure was proposed by Matas et al. to discover correspondences between image components from two images with various viewpoints. This method of removing a far-reaching number of comparing image components adds to the wide-benchmark coordinating, and it has prompted better stereo coordinating and protest acknowledgment algorithms.

Algorithm

Mathematically, the idea of Maximally Stable Regions (MSR)[7] was originally characterized in, by thinking about the set of all conceivable limits of a force of an image, *I*, to a twofold image E_t :

$$E_t(x) = \begin{cases} 1 \ ; \ I(x) \ \geq t \\ 0 \ ; \ I(x) < t \end{cases}$$

A Maximally Stable Extreme Region(MSER) [7] is then an associated region in E_t with minimal size change over several edges. An advancement that progressively builds the edge t, detects just dim regions called MSER+, brilliant regions called MSER- are acquired by modifying the forced image. The quantity of limits for which the region is stable is called the edge of the region.

A direct method to detect more regions with MSER, if a shading image is accessible, is to run the detector on the dim scale image, and on red-green and yellow-blue channels, as was finished with DoG (Difference of Gaussian) [7]. This will cause some duplicate detections. We evacuate duplicates in the additional channels if the region centroid separate is underneath 4 pixels, and the zones contrast by under 10%. The idea all the more basically can be clarified by thresholding. All the pixels underneath a given limit are 'dark' and all those above or equal are 'white'. Given a source image, on the off chance that we create an arrangement of thresholded result images I_t where each image t relates to an expanding limit t, we would see initial a white image, at that point 'dark' spots comparing to local force minima will seem then become bigger. These 'dark' spots will eventually converge until the point when the entire image is dark. The set of all associated parts in the grouping is the set of all extremal regions. In that sense, the idea of MSER is connected to one of the segment trees of the image. The segment tree without a doubt gives a simple method to actualizing MSER.

Implementation

The original algorithm of Matas et al. is $O(n \log(\log(n)))$ where n is the number of pixels here. It continues by first arranging the pixels by power. This would take $O(n)$ time utilizing BINSORT. Subsequent to arranging, pixels are set apart in the image, and the rundown of developing and combining associated parts and their territories is kept up utilizing the association discover algorithm. This would $O(n \log(\log(n)))$ time. By and by, these means are quick. Amid this procedure, the territory of each associated segment as a component of power is put away delivering an information structure. A convergence of two parts is seen as the end of the presence of the smaller segment and an inclusion of all pixels of the smaller segment into the bigger one. In the extremal regions, the 'maximally stable' ones are those comparing to edges where the relative zone change as a component of a relative difference in the edge is at the local minimum, i.e. the MSER are the parts of the image where local binarization is stable over a vast scope of edges.

The segment tree is the set of all associated segments of the edges of the image, requested by incorporation. Along these lines, this structure offers a simple route for actualizing MSER.

3.5 FAST FEATURE DETECTOR

Rather than other conventional feature detectors FAST [7] was planned for the most part for Machine Learning algorithms and a monumental increment in the speed of calculation with the direct loss of the features detected. FAST (Features from Accelerated Segment Test) feature detection algorithm was proposed by Edward Rosten and Tom Drummond in their paper "Machine learning for fast corner detection" in 2006. The most encouraging preferred standpoint of the FAST corner detector is its computational efficiency. Alluding to its name, it is quick and without a doubt, it is speedier than numerous other surely understood feature extraction methods, for example, Difference of Gaussians (DoG) utilized by the SIFT, SUSAN and Harris detectors. Besides, when machine learning strategies are connected, superior execution regarding calculation time and assets can be realized. The FAST corner detector is exceptionally appropriate for real-time video preparing application on account of this rapid execution.

Algorithm

- Select a pixel ρ in the image which is to be recognized as an interest point or not.
- Select proper edge value t.
- Consider a hover of 16 pixels around the pixel under test. This is shown in Fig 3.1.

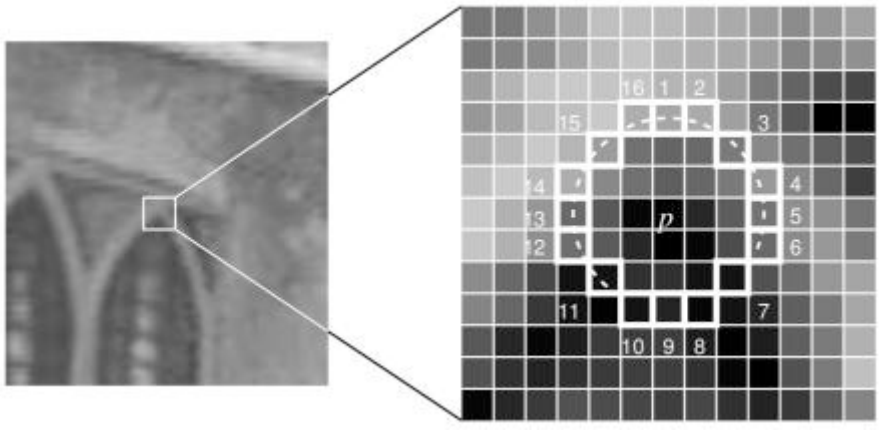

Fig 3.1 Circle of 16 points around the selected pixel ρ (source: docs.opencv.org)

- Presently the pixel ρ is a corner if there exists a set of n adjoining pixels in the hover (of 16 pixels) which are all brighter than $I_\rho + t$ or all darker than $I_\rho - t$. This appeared as white dash lines in the fig 6. The value of *n* used is 12.
- A rapid test was proposed to avoid countless corners [7]. This test looks at just the four pixels at 1, 9, 5 and 13 (First 1 and 9 are tested on the off chance that they are excessively brighter or darker. Provided that this is true, at that point checks 5 and 13). In the event that ρ is a corner, at that point, no less than three of these must all be brighter than $I_\rho + t$ or darker than $I_\rho - t$. On the off chance that neither of these is the situation, at that point ρ can't be a corner.

  The full segment test standard would then be able to be connected to the passed competitors by analyzing all pixels in the circle. This detector in itself displays elite point detections, however, there are several shortcomings:
  - It doesn't dismiss the same number of possibilities for n < 12.
  - The selection of pixels isn't optimal in light of the fact that its proficiency relies upon the requesting of the inquiries and circulation of corner appearances.
  - Consequences of rapid tests are discarded.
  - Different features detected are nearby to each other.

Initial 3 points are tended to with a machine learning approach. The last one is tended to utilizing non-maximal suppression.

Non-Maximal Suppression

Detecting different interest points in nearby areas is another issue. It is explained by utilizing Non-Maximal Suppression.

1. Figure a score γ for all the detected feature points. γ is the whole of total difference between p and 16 encompassing pixels values.
2. Consider two neighboring key points and figure their γ values.
3. Dispose of the one with the lesser γ value.

Finally, FAST feature detector is several times speedier than other existing corner detectors. Be that as it may, it isn't robust to elevated amounts of commotion. It is dependent on a limit.

3.6 HARRIS FEATURE DETECTOR

Harris Corner Detector [7] is a corner detection algorithm that is generally utilized as a part of computer vision algorithms to separate corners and derive features of an image. It was first presented by Chris Harris and Mike Stephens in 1988 upon the change of Moravec's corner detector. Contrasted with the past one, Harris' corner detector considers the differential of the corner score with reference to heading specifically, rather than utilizing moving patches for each 45-degree edges and has been turned out to be more precise in recognizing edges and corners. From that point forward, it has been enhanced and received in numerous algorithms to pre-process images for resulting applications.

Algorithm

Without losing any of the generality, we will accept a grayscale 2-dimensional image is being utilized. Let this image be given by I. Consider taking an image fix over the region (x, y) and moving it by $(\Delta x, \Delta y)$. The aggregate of squared differences (SSD) between these two patches, meant by f, is denoted by:

$$f(x,y) = \sum_{(x_k,y_k)\in W} (I(x_k,y_k) - I(x_k + \Delta x, y_k + \Delta y))^2$$

$I(x + \Delta x, y + \Delta y)$ can be determined through approximation by using Taylor's expansion. Let I_x and I_y be considered the partial derivatives of image I, with the end goal that,

$$I(x + \Delta x, y + \Delta y) \approx I(x,y) + I_x(x,y)\Delta x + I_y(x,y)\Delta y$$

This results in the final approximation of

$$f(x,y) \approx \sum_{(x,y)\in W}(I_x(x,y)\Delta x + I_y(x,y)\Delta y)^2 ,$$

which can interpret in the matrix form as

$$f(x,y) \approx \begin{pmatrix} \Delta x & \Delta y \end{pmatrix} M \begin{pmatrix} \Delta x \\ \Delta y \end{pmatrix},$$

Where *M* is considered as the structure tensor,

$$M = \sum_{(x,y)\in W} \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix} = \begin{bmatrix} \sum_{(x,y)\in W} I_x^2 & \sum_{(x,y)\in W} I_x I_y \\ \sum_{(x,y)\in W} I_x I_y & \sum_{(x,y)\in W} I_y^2 \end{bmatrix}$$

Finally comes the calculation of Harris response calculation. Here, we calculate the tiniest eigenvalue of M, the structure sensor using

$$\lambda_{min} \approx \frac{\lambda_1 \lambda_2}{(\lambda_1 + \lambda_2)} = \frac{\det(M)}{trace(M)}$$

where $trace(M) = m_{11} + m_{22}$.

CHAPTER 4
REGION SEGMENTATION

Region Segmentation [4], also known as Image segmentation is the way toward parceling an advanced image into different sections (sets of pixels, otherwise called super-pixels). The objective of segmentation is to improve and additionally change the portrayal of an image into something that is more important and less demanding to investigate. Image segmentation is commonly used to find items and limits (lines, bends, and so on.) in images. All the more accurately, image segmentation is the way toward appointing a mark to each pixel in an image to such an extent that pixels with a similar name share certain qualities.

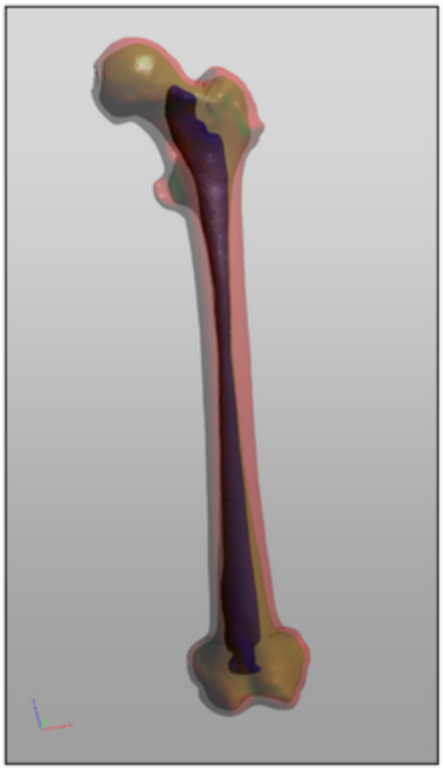

Fig 4.1 Model of segmented femur (source: wikipedia.org)

The aftereffect of image segmentation is an arrangement of portions that by and large cover the whole image or an arrangement of forms extricated from the image. Every one of the pixels in a locale is comparable as for some trademark or figured property, for example, shading, force, or surface. Nearby areas are altogether unique regarding the same characteristics. At the point when connected to a heap of images, run of the mill in medicinal imaging, the subsequent forms after image segmentation can be utilized to make 3D recreations with the assistance of interjection calculations like Marching 3D squares. Some of the very extensive list of applications regarding region segmentation includes Content-based image retrieval, Medical imaging, Machine Vision, Object Detection etc. [4]

There are multiple methods to achieve region segmentation but the two most used are Thresholding and Clustering.

4.1 THRESHOLDING

The simplest method of image segmentation is known as the thresholding method. This method is based on a clasp level (or a threshold esteem) to transform a dark scale image into a parallel image. There is also an adjusted histogram thresholding.

The key of this method is to select the threshold esteem (or values when numerous levels are selected). Several well-known methods are used in industry including the greatest entropy method, Otsu's method (most extreme difference), and k-means clustering. As of late, methods have been created to threshold computed tomography (CT) images [7]. The key thought is that, not at all like Otsu's method, the thresholds are gotten from the radiographs instead of the (reconstructed) image.

New methods suggested the usage of multi-dimensional fuzzy govern based non-straight thresholds. In these works, decision over every pixel's membership to a segment is based on multi-dimensional rules got from fuzzy rationale and transformative algorithms based on image lighting condition and application.

4.2 CLUSTERING

Cluster analysis or clustering is the task of the collection a set of objects in such a way that objects in the same gathering (called a cluster) are more similar (in some sense) to each other than to those in different groups (clusters). It is the principal task of exploratory information mining, and a typical strategy for statistical information analysis, used in numerous fields, including machine learning, design acknowledgment, image analysis, data recovery, bioinformatics, information compression, and computer graphics.

Cluster analysis itself is not one specific calculation, but rather the general task to be solved. It can be accomplished by various algorithms that contrast significantly in their idea of what constitutes a cluster and how to productively discover them. Famous notions of clusters incorporate groups with small distances between cluster members, dense areas of the information space, intervals or specific statistical distributions. Clustering can, thusly, be detailed as a multi-target enhancement issue. The proper clustering calculation and parameter settings (counting parameters such as the distance capacity to use, a density threshold or the

quantity of expected clusters) rely upon the individual informational index and proposed use of the results. Cluster analysis as such is not a programmed task, but rather an iterative process of information discovery or intuitive multi-target advancement that involves trial and disappointment.

The most widely used clustering algorithm is K- Means clustering algorithm.

4.2.1 K-Means Clustering

K-Means Clustering is an algorithm to perform vector quantization, initially from signal processing, that is well known for cluster analysis in information mining. k-means clustering primary objective is to differentiate n observations into k clusters in which each and every observation is a part of the cluster with the closed average, becoming the part of the model of the cluster. This results in a dividing of the information space.

The issue is computationally troublesome (NP-hard); in any case, due to the existence of effective heuristic algorithms that are ordinarily utilized and unite quickly to a nearby ideal. These are usually similar to the desire amplification calculation for mixtures of Gaussian distributions through an iterative refinement approach utilized by both k-means and Gaussian Mixture Modeling [4]. Moreover, they both use cluster centers to show the information; in any case, k-means clustering tends to discover clusters

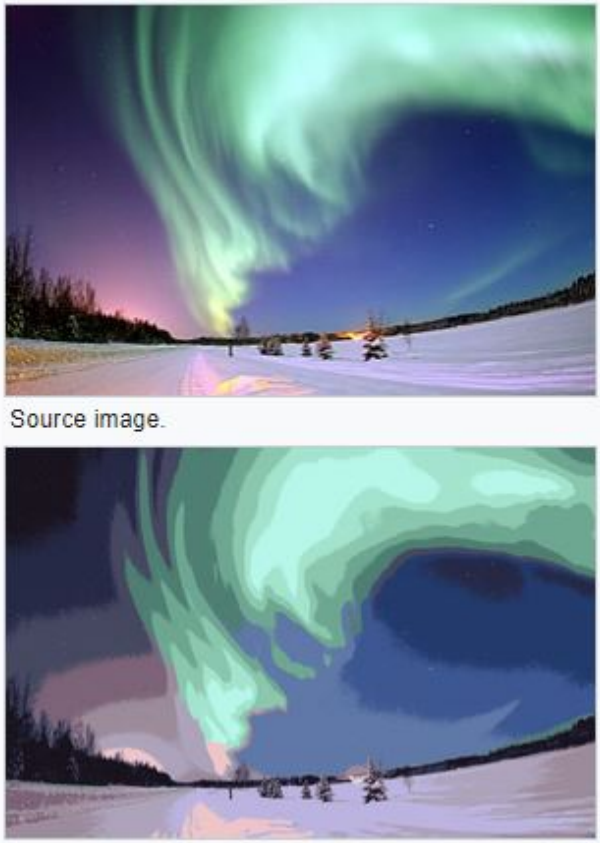

Fig 4.2 Two images showing the K-Means clustering algorithm. The topmost image is the source image and the bottom image is the resultant image with K = 8. (source: Wikipedia.org)

of similar spatial degree, while the desire boost mechanism allows clusters to have diverse shapes.

Algorithm

The Algorithm for the implementation of K – Means Clustering algorithm is as follows:

- Read the image into the program.
- Decide the number of clusters *K* to be used.
- Conversion of the RGB image, if applicable, to a grayscale image.
- Resizing of the converted grayscale image into a one-dimensional array of pixels of size *r x c.*
- Calculation of the range of intensity of the grayscale image.

$$range = [(Maximum\ intensity\ value) - (Minimum\ intensity\ value)]$$

- Calculation of the centroid value.

$$centroid\ 1 = {}^{range}/_{number\ of\ cluster}$$

$$centroid\ 2 = (2 \times centroid\ 1)$$

- Calculating the difference between the primary intensity values and the calculating the remaining centroid values.
- Depending upon the least difference gathered, segregate the intensity reading into the respective clusters.
- Finally, repeat the step w.r.t. the calculation of centroid value but this time for,

  $$centroid\ 3 = (3 \times centroid\ 1)$$

  and therefore,

  $$centroid\ n = (n \times centroid\ 1)$$

CHAPTER 5
BAG-OF-FEATURES

A Bag-of-Features is training model incorporated from the field of Natural Language Processing (NLP) to be used on the image classification. Traditionally, bag-of-features is derived from bag-of-words model. The bag-of-words demonstrates a streamlining portrayal utilized as a part of regular dialect handling and data recovery (IR) [9]. It is also called as the vector space display. In this model, content, for example, a sentence or an archive is transformed as the bag (multiset) of its words, dismissing language structure and even word arrangement, however, keeping assortment safe. The bag-of-words demonstration has likewise been utilized for computer vision. A bag-of-words model is generally utilized as a part of strategies for report characterization where the recurrence of an event of each word is utilized as a feature for preparing a classifier.

Coming to the implementation part of the project, in computer vision, a bag-of-words model (BoW) can be connected to image classification, by regarding image features as words. It is then considered as a Bag-of-Features(BoF) [9]. In archive classification, a bag-of-words is a meager vector of event tallies of words; that is, an inadequate histogram is formed over the vocabulary. In computer vision, a bag-of-features is a vector of event checks of a vocabulary of nearby image features.

5.1 REPRESENTATION OF AN IMAGE

To describe an image utilizing the bag-of-features demonstrates that an image can be dealt with as a record. Correspondingly, "words" in images should be explained as well. To accomplish this, it, as a rule, incorporates following three stages: feature detection, feature description, and generation of a codebook. A meaning of the bag-of-features model can be the "histogram portrayal based on free features". Content-based Image Indexing and Recovery (CBIR) seems, by all accounts, to be the early adopter of this image portrayal method.

5.2 REPRESENTATION OF FEATURES

After the step of feature detection, each image is divided by a few local fixes. Features portray strategies that manage how to detect the patches as numerical vectors. These vectors are called feature descriptors. A decent descriptor is ought to be able to deal with the intensity, scale, rotation and relative varieties to some degree. A standout amongst the most well-known descriptors is Speeded Up Robust Features (SURF) which is used throughout this project. SURF changes each patch of an image to a 128-dimensional vector. After this progression, each image is a gathering of vectors of a similar measurement (128 for SURF), where the request of various vectors is of no significance.

5.3 GENERATION OF A CODEBOOK

The last advance for the bag-of-features model is to change over the vector-represented patches to "codewords", similar to words in content archives, which likewise creates a "codebook", closely resembling a word lexicon. A codeword can be considered as an agent of a few comparative patches. One basic technique is performing K-Mean clustering over every one of these vectors. Codewords [9] are then characterized as the focuses of the educated clusters. The quantity of the clusters is the codebook measure that is practically equivalent to the span of the word lexicon.

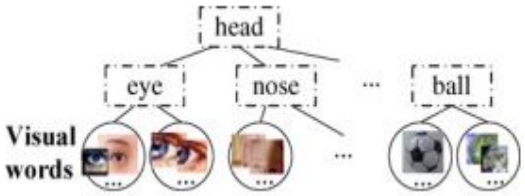

Fig 5.1 Flowchart displaying the visual word/ features extracted. [8]

Going with this, every patch in an image is mapped to a specific codeword through the bunching procedure and afterward, the image can now be represented using the histogram of the ensued codewords.

5.4 RECOGNITION BASED ON BoF Model

Computer vision specialists have built up a few learning strategies to use the bag-of-features to display image related assignments, for example, object classification. These techniques can generally be separated into two classes, generative and discriminative models. For various label arrangement issues, the confusion matrix can be utilized as an assessment metric.

5.6 CONFUSION MATRIX

When considering the vast field of machine learning and particularly the issue of statistical classification, a confusion matrix [9], also known as an error matrix, is a particular table design that permits representation of the execution of an algorithm, commonly a supervised learning algorithm. Each row of the matrix speaks to the cases in an anticipated class while every segment speaks to the examples in a real class and the other way around. The name comes from the way that it makes it simpler to check whether the framework is befuddling two classes, i.e., ordinarily mislabeling one as another.

It is an exceptional sort of contingency table, with two measurements, "real" and "anticipated", and indistinguishable arrangements of "classes" in the two measurements. For an example, we take a classification system that can distinguish between three categories of images namely cats, rabbits and dogs. The generation of a confusion matrix will confine the results of the algorithm used for any future validation or any quantitative analysis. Taking a sample of 27 pictures, 8 are cats, 13 are rabbits and 6 are dogs.

		Actual class		
		Cat	Dog	Rabbit
Predicted class	Cat	5	2	0
	Dog	3	3	2
	Rabbit	0	1	11

Fig 5.2 Confusion matrix for three categories (source: wikipedia.org)

Here, in the presented confusion matrix, of the true 8 cats, the algorithm's output was that three were dogs. Out of the six dogs, the algorithm predicted that one of them was a rabbit and the remaining two were cats. Hence, looking at the result of this confusion matrix, we can distinguish that the algorithm used in this case has some trouble differentiating between dogs and cats but is clearly able to make the difference between rabbits and the other two categories of animals. Furthermore, all the correct predictions are on the diagonal of the confusion matrix making it relatively easy to distinguish the error rate as they will be the values other that the diagonal.

CHAPTER 6
SUPPORT VECTOR MACHINE (SVM)

A Support Vector Machine (SVM) [2] is a machine learning algorithmic classifier that differentiates the formal characterization by using a separating hyperplane. At the end of the day, using a given marked training information also known as supervised learning, the algorithm yields an ideal hyperplane which sorts new illustrations. In two-dimensional area, this hyperplane is defined as a line that separates a plane into two sections in which each class or a category of images lie on each side of the plane. Given an arrangement of training cases, each set is parted as having a place with either of two classifications. The SVM then prepares the calculation constructs for a model that doles out new cases to one class or the other, making it a non-probabilistic paired direct classifier.

An SVM model is a portrayal of the cases/categories as focuses in space, mapped with the goal that the cases of the different classes are partitioned by an unmistakable well-defined gap that is as wide as could reasonably be expected. New illustrations are then plotted in the same defined space and anticipated to have a place within a class in light of which side of the plane they fall.

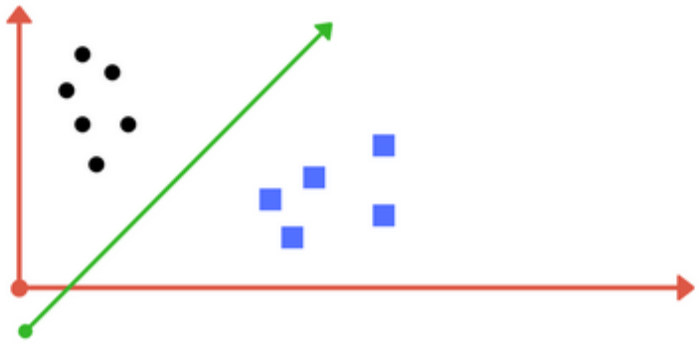

Fig 6.1 A 2-D hyperplane state system with on single hyperplane (source: medium.com)

6.1 LINEAR SVM

Linear SVM [2] is the most up to date, quick machine learning (information mining) algorithm used for taking care of multiclass classification issues from ultra-substantial datasets that actualize a unique exclusive adaptation of a cutting plane calculation for planning a straight support vector machine. Linear SVM is a straight adaptable routine implying that it makes an SVM demonstrate in a CPU time which scales directly with the span of the preparation dataset. A hyperplane is a subspace whose measurement is one not as much as that of its encompassing space. On the off chance that space is 3-dimensional then its hyperplanes are the 2-dimensional

planes, while if space is 2-dimensional, its hyperplanes are the 1-dimensional planes. This thought can be utilized as a part of any broad space in which the idea of the measurement of a subspace is characterized. Any hyperplane can be characterised by using a set of points $\underset{x_1}{\rightarrow}$ that satisfies

$$\underset{w}{\rightarrow}.\underset{x}{\rightarrow} - b = 0,$$

where $\underset{w}{\rightarrow}$ is considered as the normal vector to hyperplane. Furthermore, the parameter, $\frac{b}{mod(\underset{w}{\rightarrow})}$ is responsible for the change regarding the hyperplane w.r.t. the origin along $\underset{w}{\rightarrow}$.

In various settings, the articles which are hyperplanes may have diverse properties. For example, a hyperplane of an n-dimensional relative space is a level subset with measurement n − 1. By its temperament, it isolates the space into two identical half-spaces.
Given a dataset for training of *n* points of the form,

$$(\underset{x_1}{\rightarrow},\underset{y_1}{\rightarrow}) , \dots , (\underset{x_n}{\rightarrow}, y_n)$$

where the values of y_i range from *1* to *-1*, both of them indicating to the class to whom the point $\underset{x_i}{\rightarrow}$ affirms. Each of the point $\underset{x_i}{\rightarrow}$ can be described a ρ – dimensional vector. The main objective is to gather the "maximum – margin of the hyperplane" that is able to differentiate between a group of point like $\underset{x_i}{\rightarrow}$ such that $y_i = 1$, from the set of points for which $y_i = -1$. This is characterised as the distance between the said hyperplane and the closest set of points $\underset{x_i}{\rightarrow}$, from any of the group/category which is then maximised.

6.1.1 Hard Margin

On the off chance that the preparation information is directly distinguishable, we can choose two parallel hyperplanes that different the two classes of information, so the separation between them is as expansive as could be expected under the circumstances. The district limited by these two hyperplanes is known as the "margin" [2], and the most extreme margin plane is the hyperplane that untruths between them. With

appropriate dataset rescaling these hyperplanes can be portrayed by the accompanying conditions:

$\underset{w}{\rightarrow}.\underset{x}{\rightarrow} - b = 1$ (any feature / data point above this equation boundary is part of class A)

and,

$\underset{w}{\rightarrow}.\underset{x}{\rightarrow} - b = -1$ (any feature / data point below this equation boundary is part of class B)

Considering the geometry aspect, the actual distance in-between any two hyperplanes is $2/mod(\underset{w}{\rightarrow})$, resulting in the maximisation of the distance between the hyperplanes that we want to minimise.

6.1.2 Soft Margin

To incorporate the cases of data that cannot be separated linearly, the concept of soft margin is extended to the SVM using the hinge loss function,

$$max\ (\ 0{,}1 - \ y_i\ (\vec{w}\ .\ \vec{x} - b\ )).$$

The resultant of this function is null if the said constraint is fulfilled, or, if $\vec{x_i}$ lies on the right side of the margin induced by the hyperplane. However, if the data is on the wrong side of the margin, in that case, the value of the function is directly proportional to the actual distance taken from the margin.

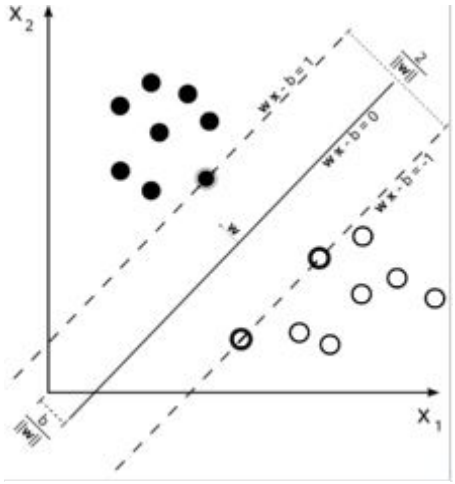

Fig 6.2 2-D single hyperplane showing hard and soft margins. (source: wikipedia.org)

CHAPTER 7

RESULTS AND ANALYSIS

This section constitutes of all the experimental observations gathered during the successful compilation of all the algorithm used in the project as well as mentioned in the report. These algorithms are SURF, MSER, FAST [7] and Harris Feature Detectors; K-Means clustering algorithm [3] for region segmentation; bag-of-features for the visual word generation and finally Support Vector Machine(SVM) for the training of the dataset.

SURF FEATURE DETECTOR

In the both the images taken as an example, the SURF detector seems to detect MOST of the important features necessary to describe the image. Quantitatively, SURF was able to detect 315 features in Image1 and 1062 features in Image2 which is quite good for a detector.

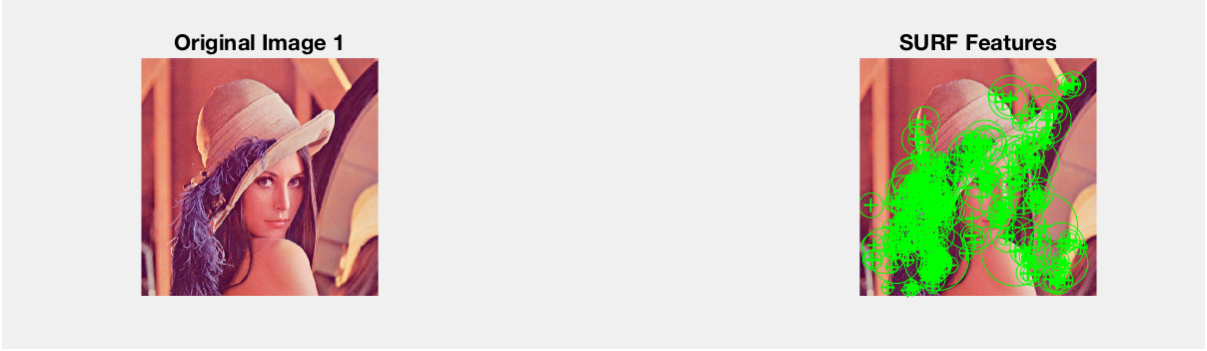

Fig 7.1 SURF detector used in MATLAB in Image1[9]

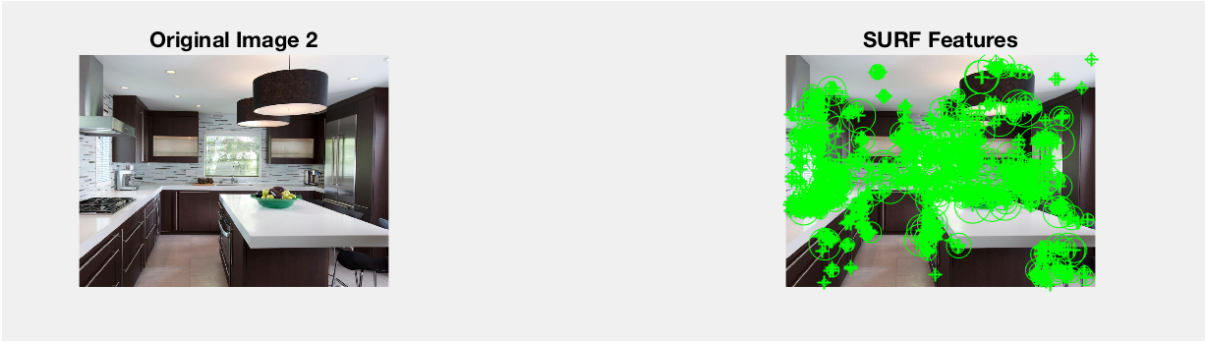

Fig 7.2 SURF detector used in MATLAB in Image2[10]

MSER FEATURE DETECTOR

In the both the images taken as an example, the MSER detector seems to detect ALL of the important features necessary to describe the image. Quantitatively, MSER was able to detect 354 features in Image1 and 1405 features in Image2.

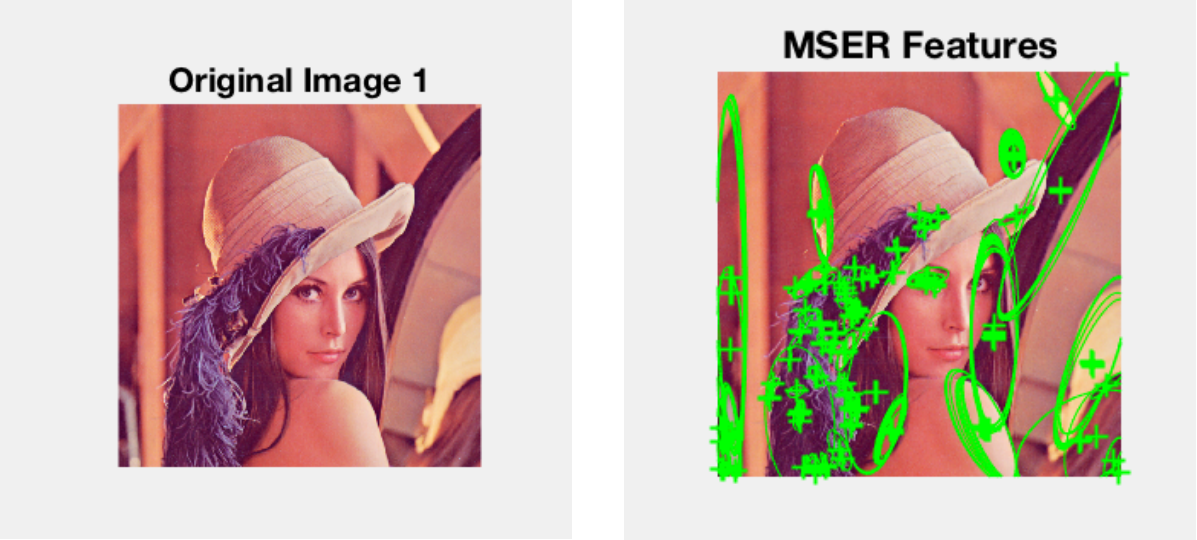

Fig 7.3 MSER detector used in MATLAB in Image1[9]

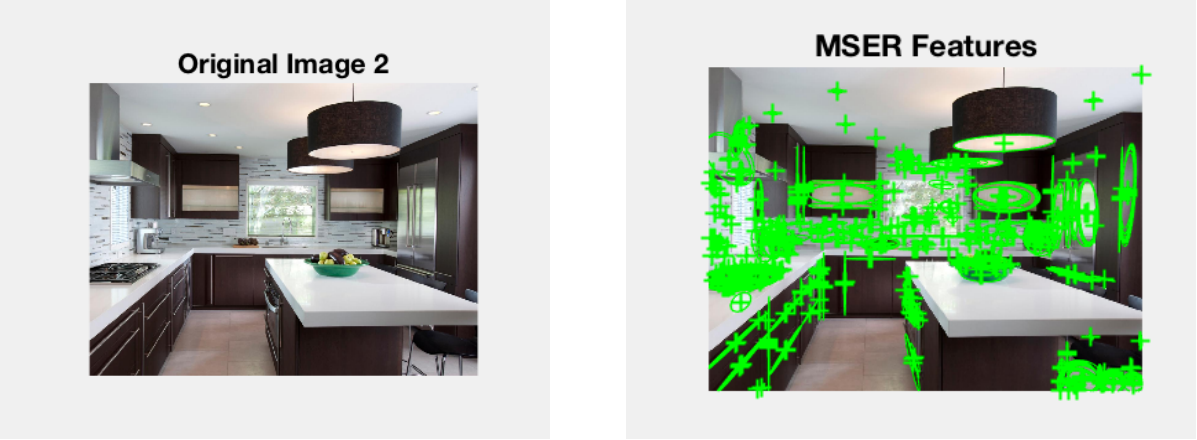

Fig 7.4 MSER detector used in MATLAB in Image2[10]

FAST FEATURE DETECTOR

In the both the images taken as an example, the FAST detector seems to detect VERY FEW of the important features necessary to describe the image. Quantitatively, FAST was able to detect 153 features in Image1 and 499 features in Image2.

Fig 7.5 FAST detector used in MATLAB in Image1[9]

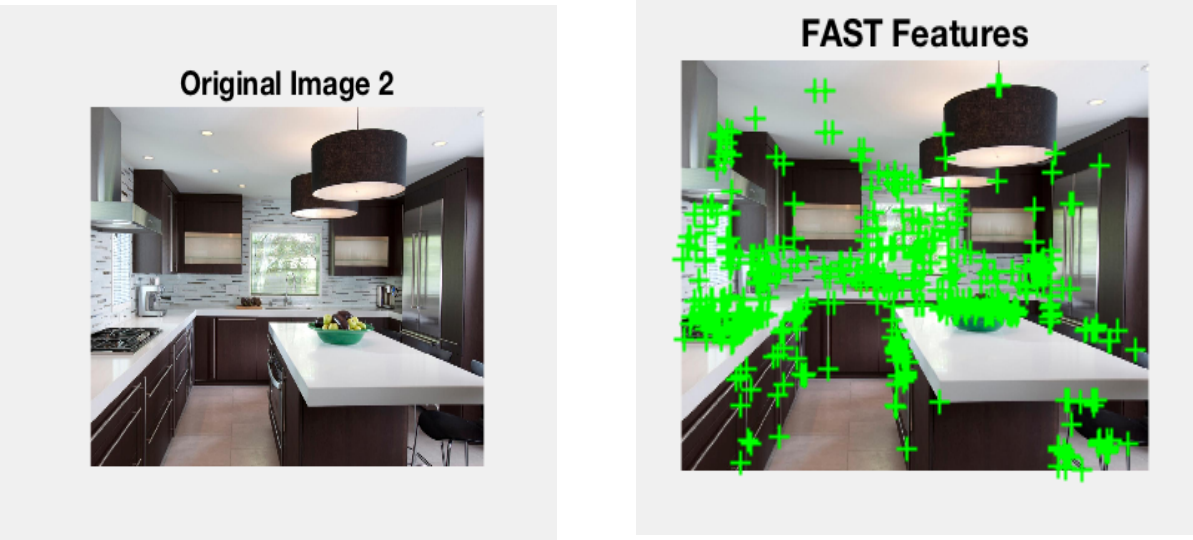

Fig 7.6 FAST detector used in MATLAB in Image2[10]

HARRIS FEATURE DETECTOR

In the both the images taken as an example, the Harris detector seems to detect a MODERATE amount of the important features necessary to describe the image. Quantitatively, Harris was able to detect 459 features in Image1 and 960 features in Image2.

Fig 7.7 Harris detector used in MATLAB in Image1[9]

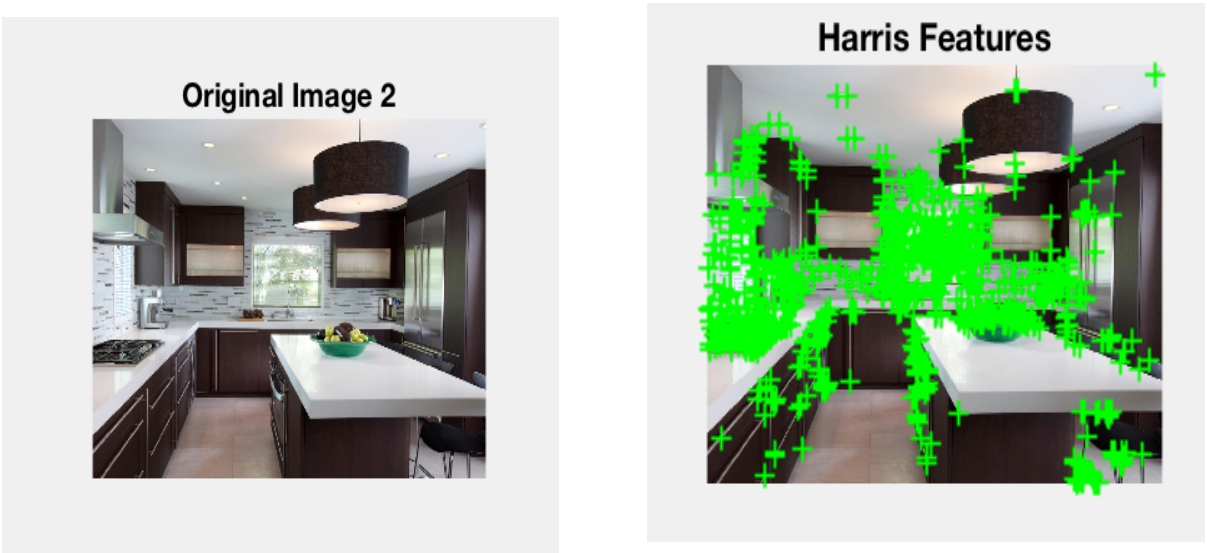

Fig 7.8 Harris detector used in MATLAB in Image2[10]

K-MEANS CLUSTERING ALGORITHM

I applied K means clustering algorithm to two images Image1 and Image2 to show the variety of clustering on different images. The number of clusters were chosen to be as K=3, the standard. In Image1 of a tissue, in can be seen that different type of tissues denoted by their different 'dye' color is successfully separated using this algorithm. In Image2, different parts of this kitchen are separated and it is worth noting here that the single object of apples is isolated really well.

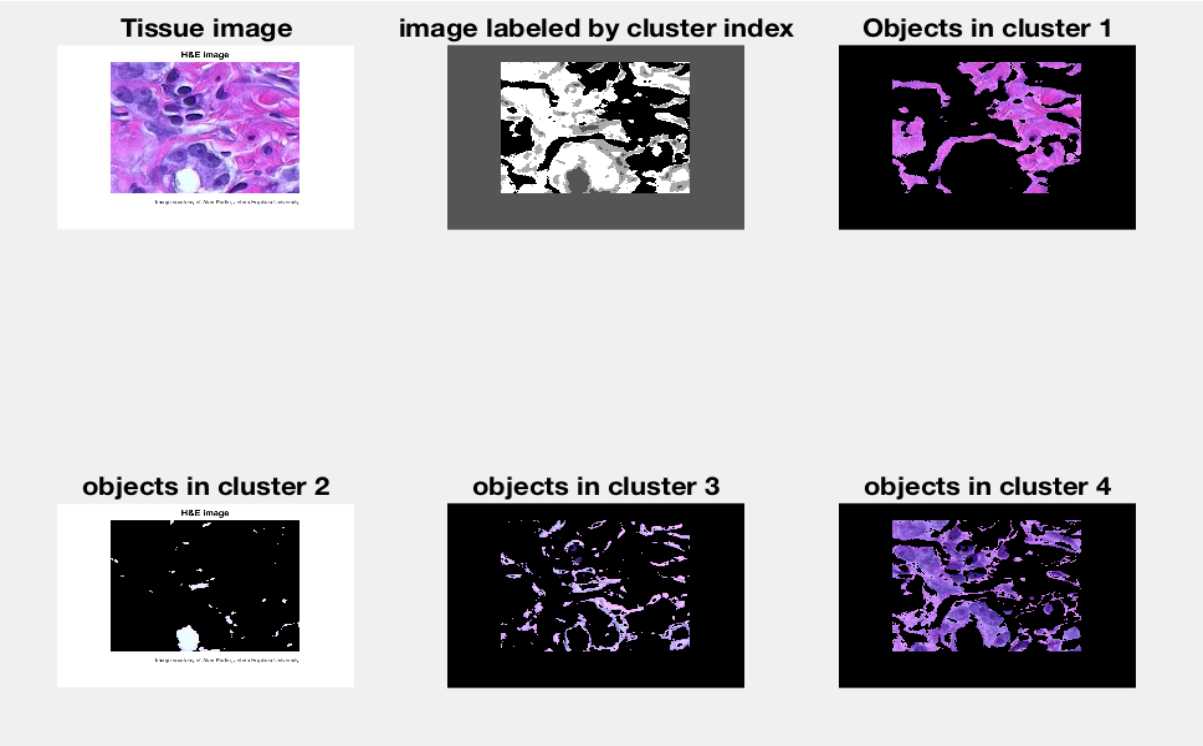

Fig 7.9 The "HESTAIN" image demonstrating K-Means Clustering [11]

Fig 7.10 The "KITCHEN" image demonstrating K-Means Clustering [10]

Looking back, a quantitative analysis was done between the four major Feature Detectors used in the real world namely SURF, MSER, FAST and Harris. Out of all the four, SURF was the clear choice. Although it didn't always detect the maximum number of features when compared to MSER but it didn't have that 3 second lag while computing as there was in the case of MSER albeit MSER detected the maximum number of features in a given image.

Region Segmentation with the help of the K-Mean Clustering was done and it's worth noting that in the segmentation of Image2, a bunch of apples are isolated with is the after all the main goal of the project.

Below is the quantitative comparison of all the feature extractors.

	SURF Features	MSER Features	FAST Features	Harris Features
Image 1	305	354	153	459
Image 2	1062	1405	499	960
Image 3	674	581	652	815
Image 4	266	489	204	142
Image 5	1058	1640	798	782

Table 7.1: Quantitative comparison between various feature extractors on five images

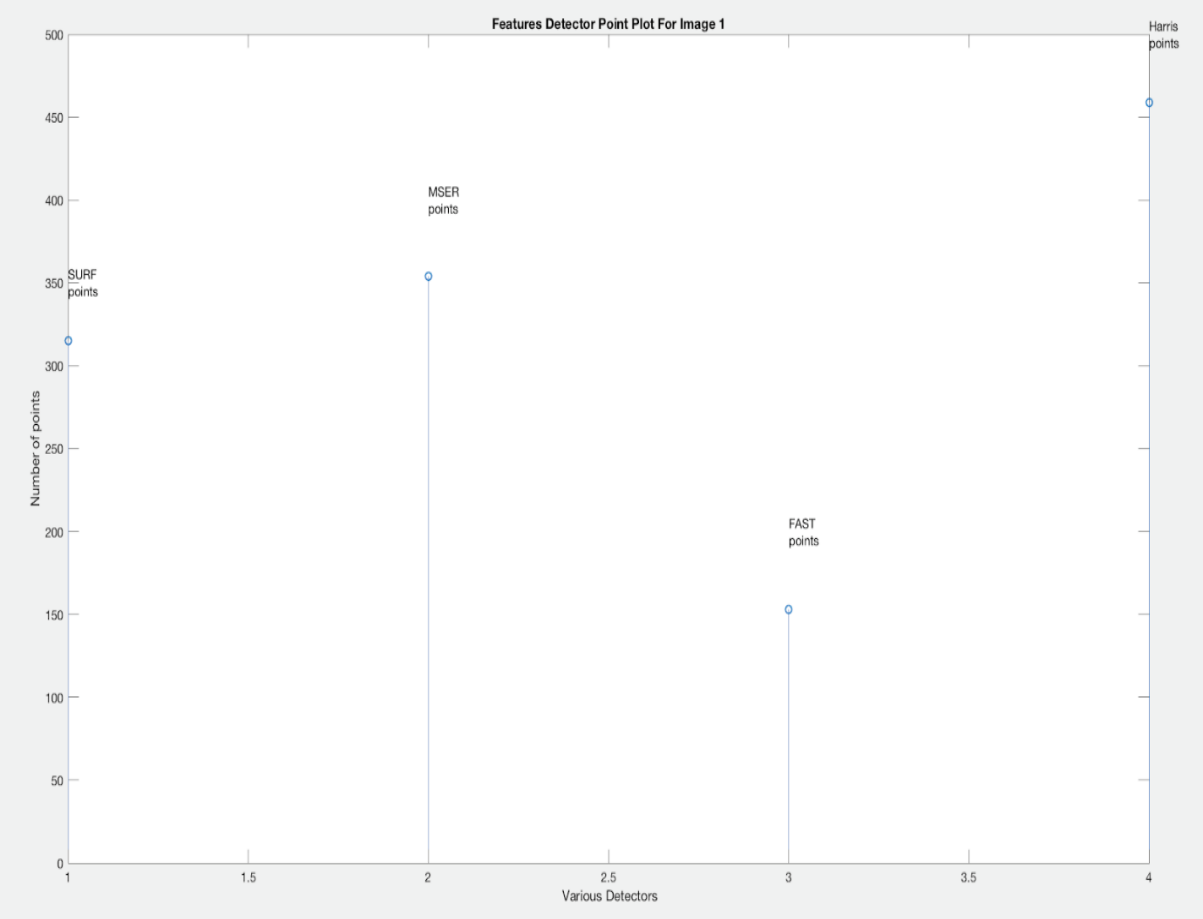

Fig 7.11 Quantitative Analysis of various feature detectors on Image1

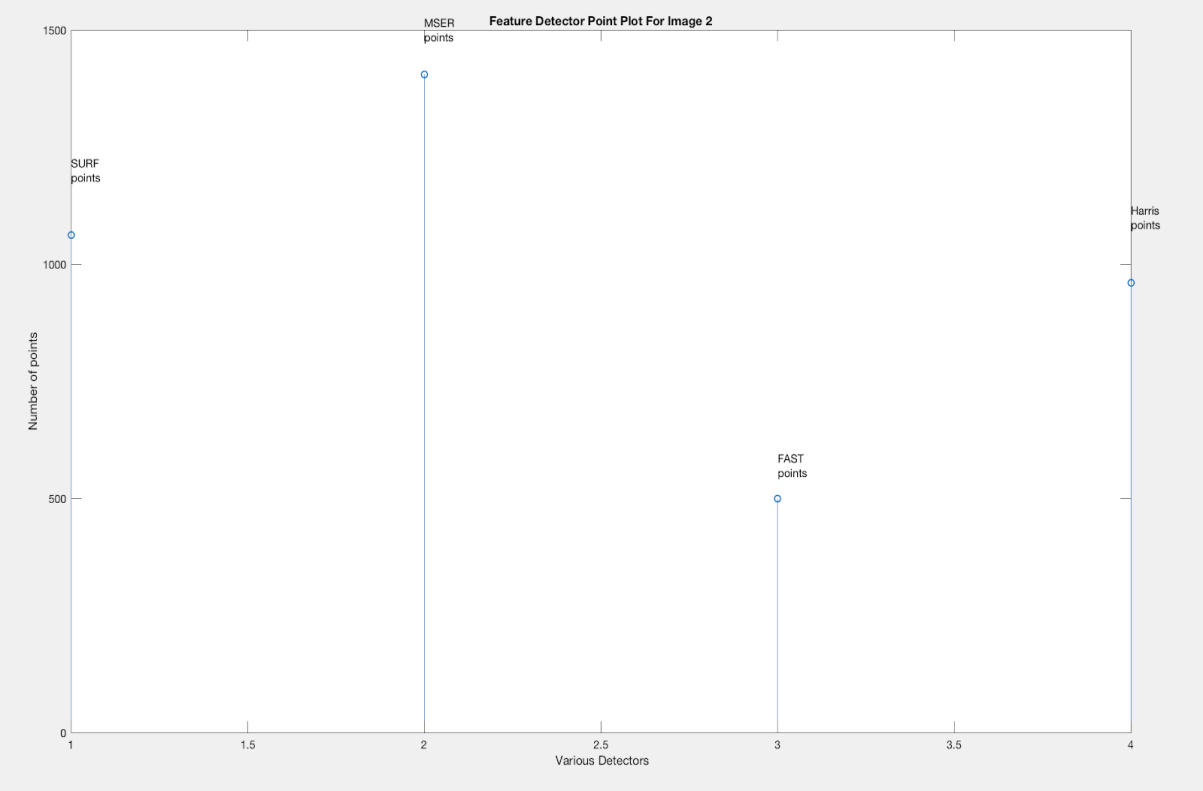

Fig 7.12 Quantitative Analysis of various feature detectors on Image2

Below is the summation of all the work done cumulatively on Feature Detection Algorithms.

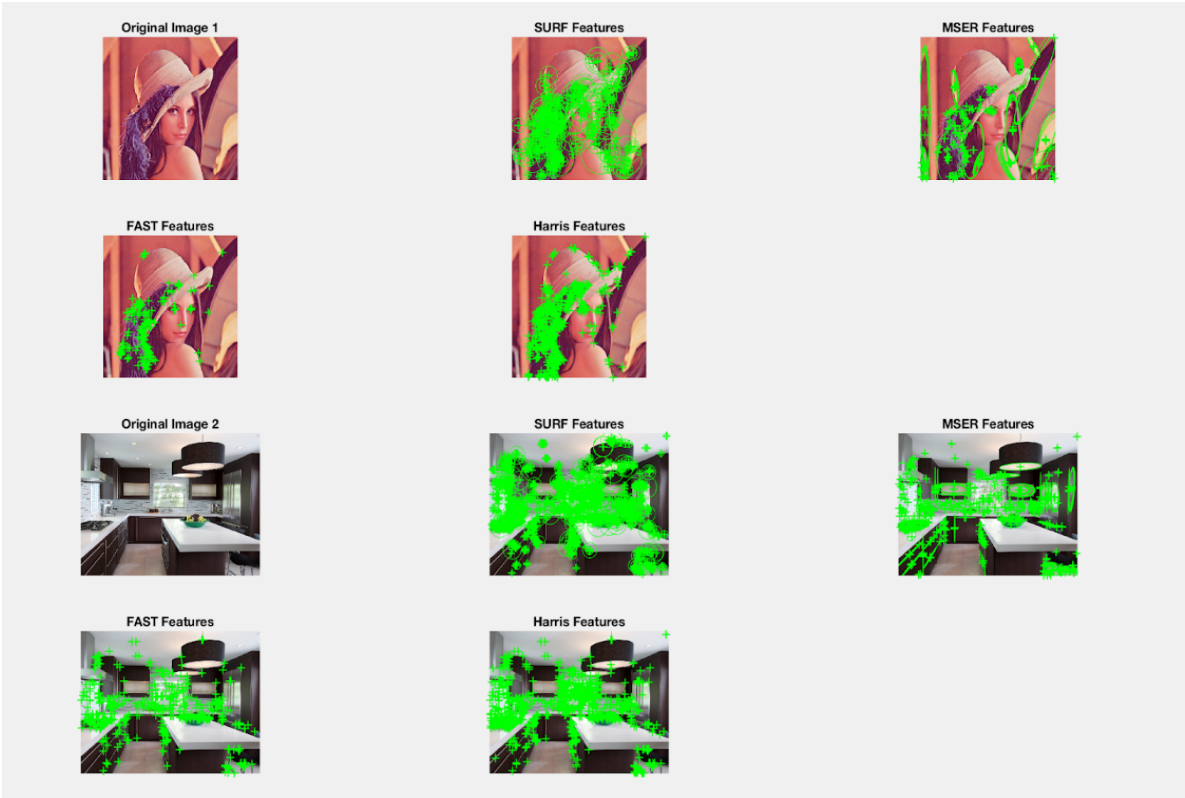

Fig 7.13 Cumulative work done on Feature Detection and their comparison on two sample images

Furthermore, below is the combined work on SURF feature detector along with the K-Means Clustering algorithm.

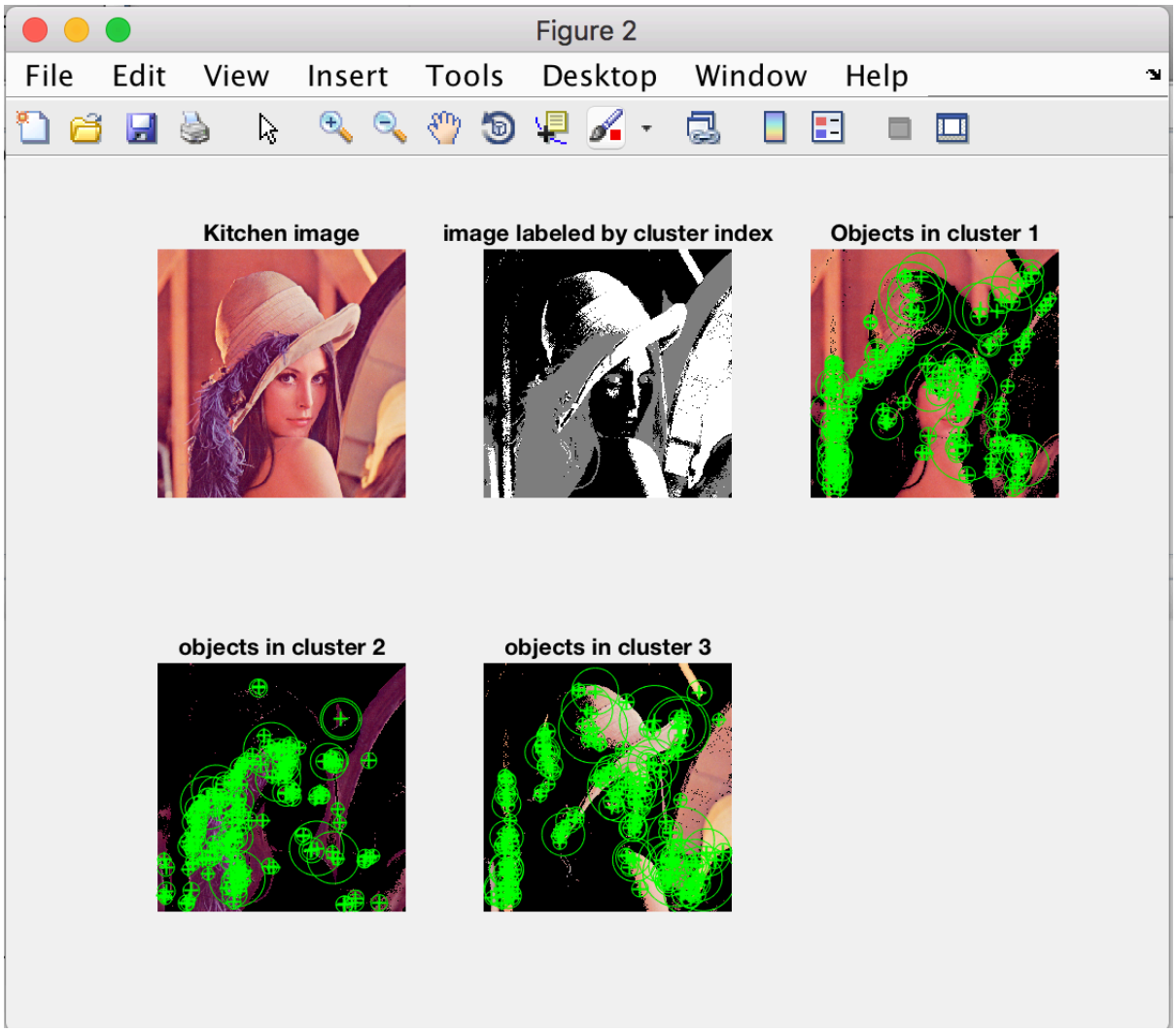

Fig 7.14 Cumulative work of SURF features along with K-Means Clustering on Image1[9]

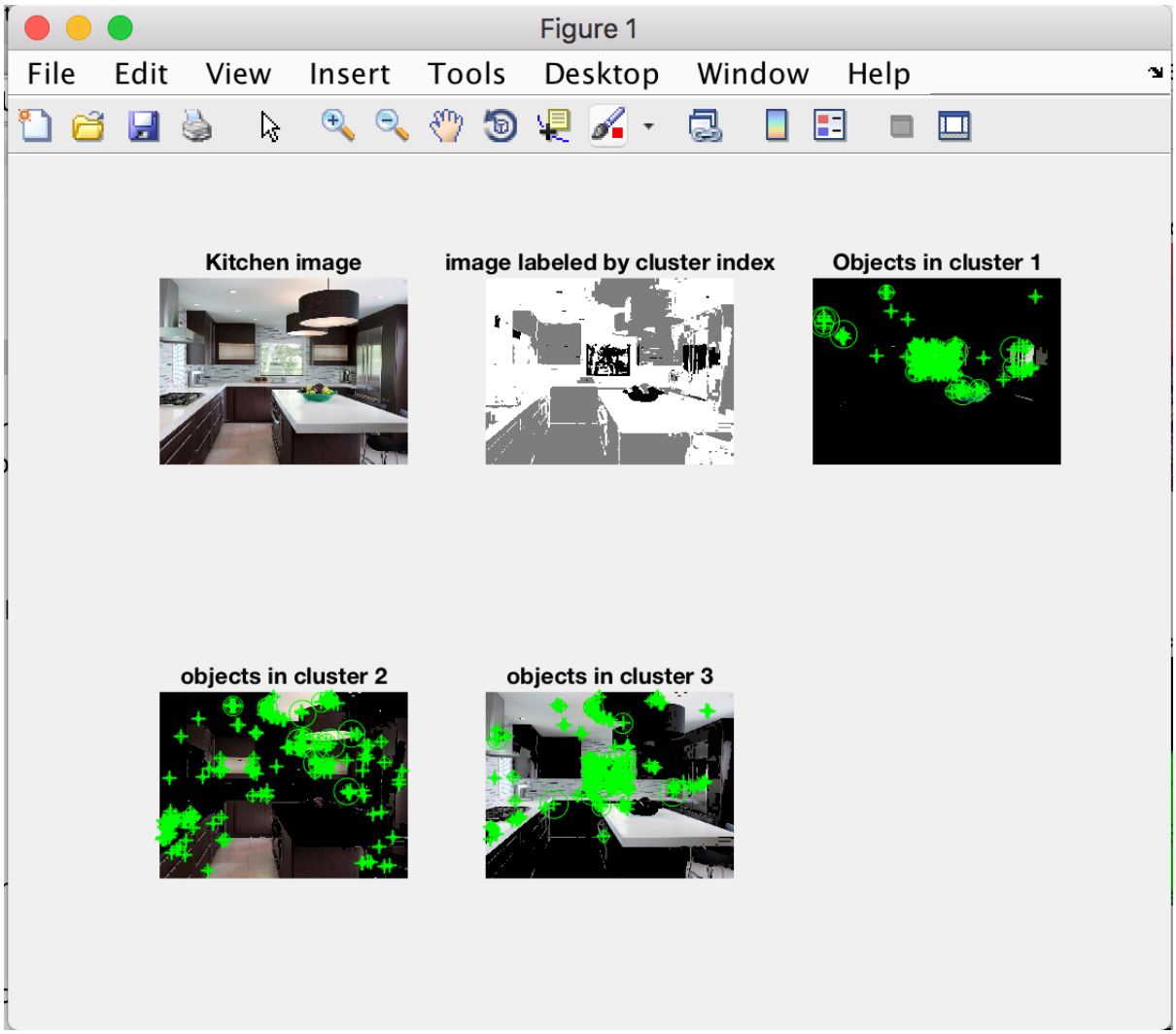

Fig 7.15 Cumulative work of SURF features along with K-Means Clustering on Image2[10]

THE TRAINING PROCESS

First up is the creation of the bag-of-features for six categories all the while the creation of their SURF points, the extraction of features, balancing of the features, keeping the 80% strongest features. After the feature detection is done, K-Means clustering is done using 500 clusters.

```
Creating Bag-Of-Features.
-------------------------
* Image category 1: airplanes_side
* Image category 2: background
* Image category 3: cars
* Image category 4: cars_bg
* Image category 5: faces
* Image category 6: motorbikes_side
* Selecting feature point locations using the Grid method.
* Extracting SURF features from the selected feature point locations.
** The GridStep is [8 8] and the BlockWidth is [32 64 96 128].

* Extracting features from 1350 images...done. Extracted 17863604 features.

* Keeping 80 percent of the strongest features from each category.

* Balancing the number of features across all image categories to improve clustering.
** Image category 2 has the least number of strongest features: 354099.
** Using the strongest 354099 features from each of the other image categories.

* Using K-Means clustering to create a 500 word visual vocabulary.
* Number of features          : 2124594
* Number of clusters (K)      : 500

* Initializing cluster centers...100.00%.
* Clustering...completed 27/100 iterations (~19.68 seconds/iteration)...converged in 27 iterations.

* Finished creating Bag-Of-Features
```

Fig 7.16 Creation of bag-of-features

Next is the graphical representation of the visual feature index.

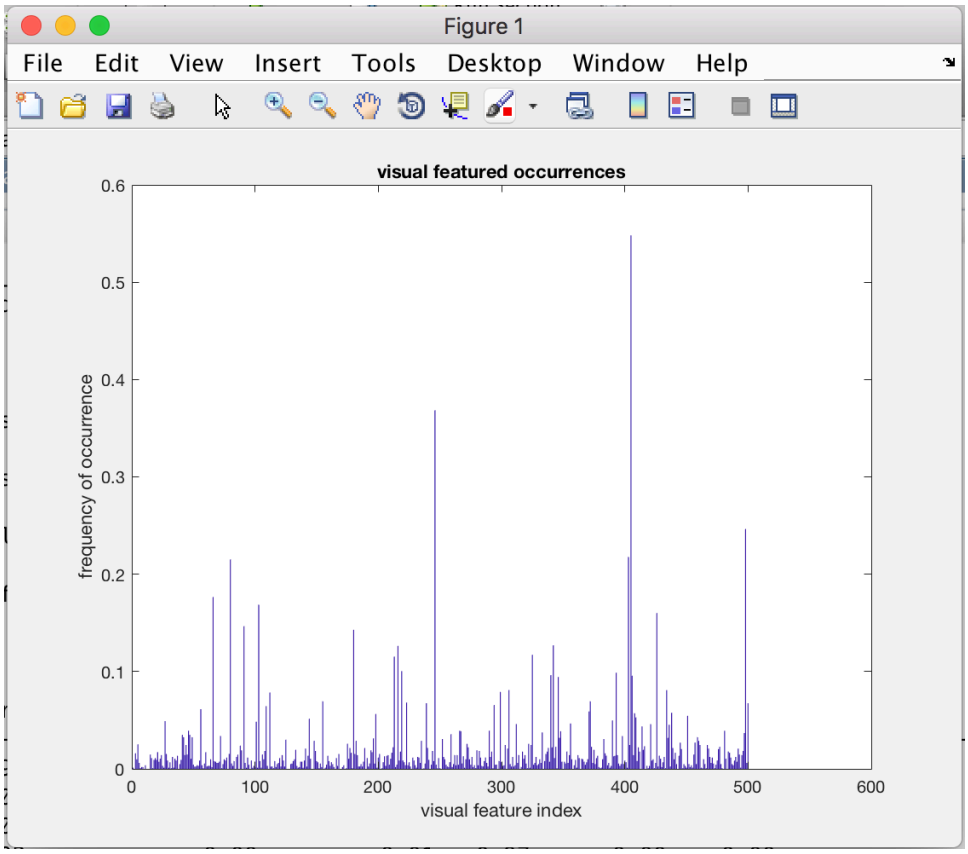

Fig 7.17 Graph showing Visual Feature Occurrence

Here, 1350 images from the training set are chosen to train the SVM.

```
Training an image category classifier for 6 categories.
--------------------------------------------------------
* Category 1: airplanes_side
* Category 2: background
* Category 3: cars
* Category 4: cars_bg
* Category 5: faces
* Category 6: motorbikes_side

* Encoding features for 1350 images...done.

* Finished training the category classifier. Use evaluate to test the classifier on a test set.
```

Fig 7.18 Training dataset used for SVM

Below is the sample test to determine the accuracy of the algorithm for four random images.

```
the input image is airplanes and the output: airplanes_side

The input image is cars and the output image is: cars

The input image is faces and the output image is: faces

The inout image is motorbike and the output image is: motorbikes_side
```

Fig 7.19 Checking for the accuracy for random images

Here is the creation of the confusion matrix and the calculations for the accuracy of the SVM against the training dataset. Used here are the rest of 1350 images from the validation set.

```
Evaluating image category classifier for 6 categories.
-------------------------------------------------------

* Category 1: airplanes_side
* Category 2: background
* Category 3: cars
* Category 4: cars_bg
* Category 5: faces
* Category 6: motorbikes_side

* Evaluating 1350 images...done.

* Finished evaluating all the test sets.

* The confusion matrix for this test set is:

                                                PREDICTED
KNOWN              | airplanes_side   background   cars   cars_bg   faces   motorbikes_side
-------------------------------------------------------------------------------------------
airplanes_side     | 0.99             0.00         0.00   0.00      0.00    0.01
background         | 0.03             0.96         0.00   0.00      0.00    0.00
cars               | 0.00             0.00         0.99   0.00      0.00    0.00
cars_bg            | 0.01             0.00         0.00   0.99      0.00    0.00
faces              | 0.01             0.00         0.00   0.00      0.97    0.02
motorbikes_side    | 0.03             0.02         0.01   0.00      0.01    0.94

* Average Accuracy is 0.97.
```

Fig 7.20 Confusion Matrix for training dataset

Finally, below is the confusion matrix of the validation image set for the SVM.

```
* Category 1: airplanes_side
* Category 2: background
* Category 3: cars
* Category 4: cars_bg
* Category 5: faces
* Category 6: motorbikes_side

* Evaluating 1350 images...done.

* Finished evaluating all the test sets.

* The confusion matrix for this test set is:

                                                PREDICTED
KNOWN              | airplanes_side   background   cars   cars_bg   faces   motorbikes_side
-------------------------------------------------------------------------------------------
airplanes_side     | 0.97             0.00         0.00   0.00      0.00    0.03
background         | 0.01             0.93         0.03   0.01      0.02    0.00
cars               | 0.00             0.00         0.99   0.01      0.00    0.00
cars_bg            | 0.02             0.01         0.01   0.96      0.00    0.00
faces              | 0.00             0.01         0.00   0.00      0.98    0.01
motorbikes_side    | 0.03             0.00         0.00   0.00      0.02    0.94

* Average Accuracy is 0.96.
```

Fig 7.21 Confusion Matrix for validation dataset

Summarising the training process, a total of 6 categories namely airplanes_side, background, cars, cars_bg, faces, and motorbikes_side contained a total of 2700+ images.

- Total features extracted were 17863604.
- 80% of the strongest features were selected and to improve the clustering, balancing of the features was done.
- Image category 2, i.e., backgrounds, contained the most number of strongest features.
- After the feature detection is over, 2124594 features along with 500 cluster centres are used for the K-Means Clustering.
- The clustering was completed in a total of 27 iterations with each iteration taking approximately 20 sec.
- Coming to the training part of the algorithm, 1350 images from the trainingSet were used.
- To show the accuracy of the program, four random images were chosen to evaluate the algorithm prematurely in Fig 7.19.
- Under evaluation of algorithm, first the trainingSet itself was used to check whether the SVM had been correctly trained or not. The results were positive with an average accuracy of 97%. Considering individual categories, airplanes_side, cars and cars_bg had the highest accuracy of 99% while the category motorbike_side had the lowest at 94%.
- Finally, for the actual validation, the rest of 1350 images from validationSet were chosen and a confusion matrix was drawn. The results were positive with an average accuracy of 96% almost equal to the accuracy of the trainingSet. Considering individual category cars had the highest accuracy of 99% while the category background had the lowest at 94%.

Coming to the statistical analysis of the project, three parameters were used besides accuracy, Precision, Recall and Fmeasure. Mathematically, Precision is the ratio of documents retrieved that are important to the query used. In layman terms, Precision is the measure that decides how valued are the said results.

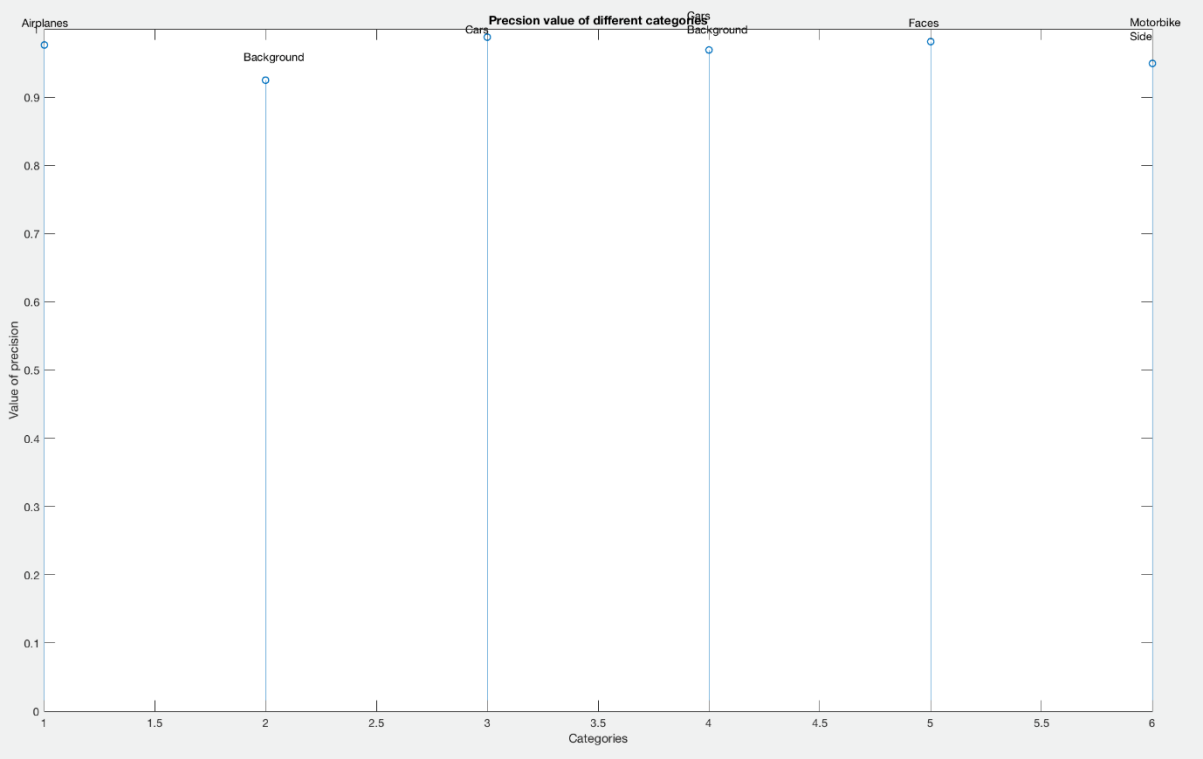

Fig 7.22 Precision value of different categories

Next measure is Recall. Mathematically, Recall is the ratio of documents retrieved verses the query asked. In layman terms, recall decides how full/complete the results are.

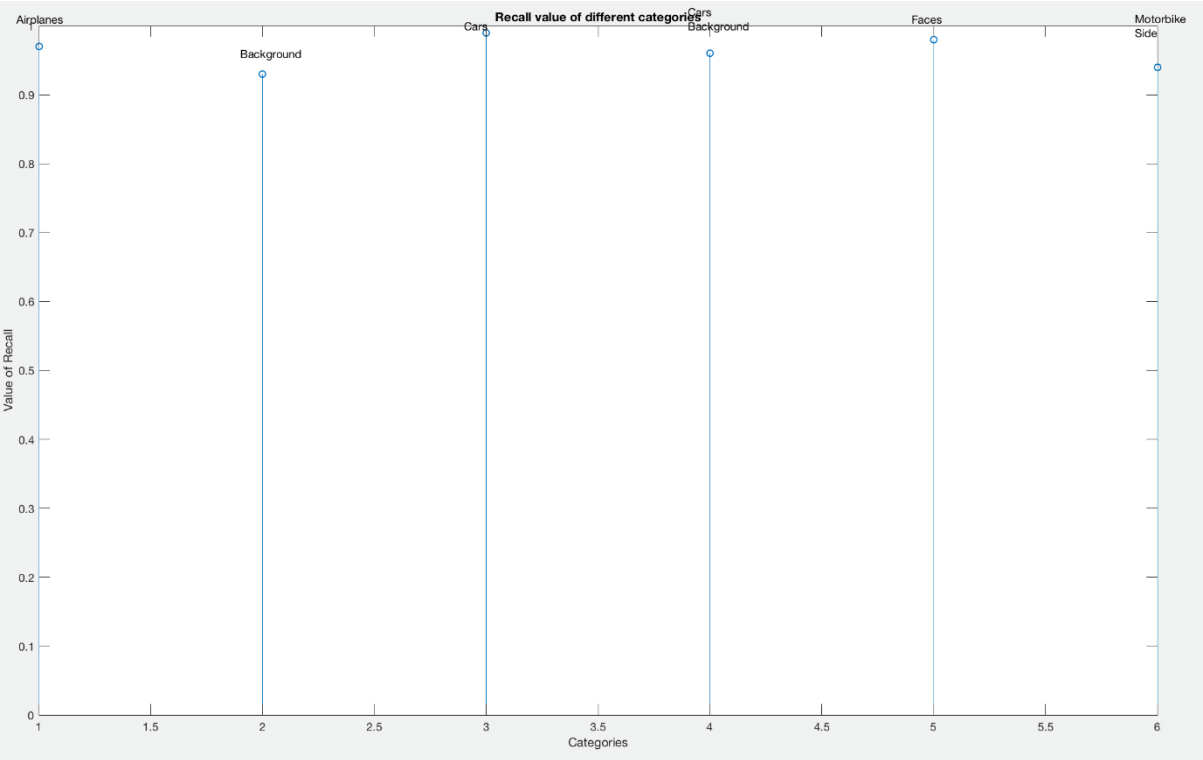

Fig 7.23 Recall value of different categories

Finally, after the calculation of both Precision and Recall, comes the Fmeasure. Fmeasure is calculated to summarise Recall and Precision. It is a summarised performance measure.

$$Fmeasure = \frac{2}{\frac{1}{Precision} + \frac{1}{Recall}}$$

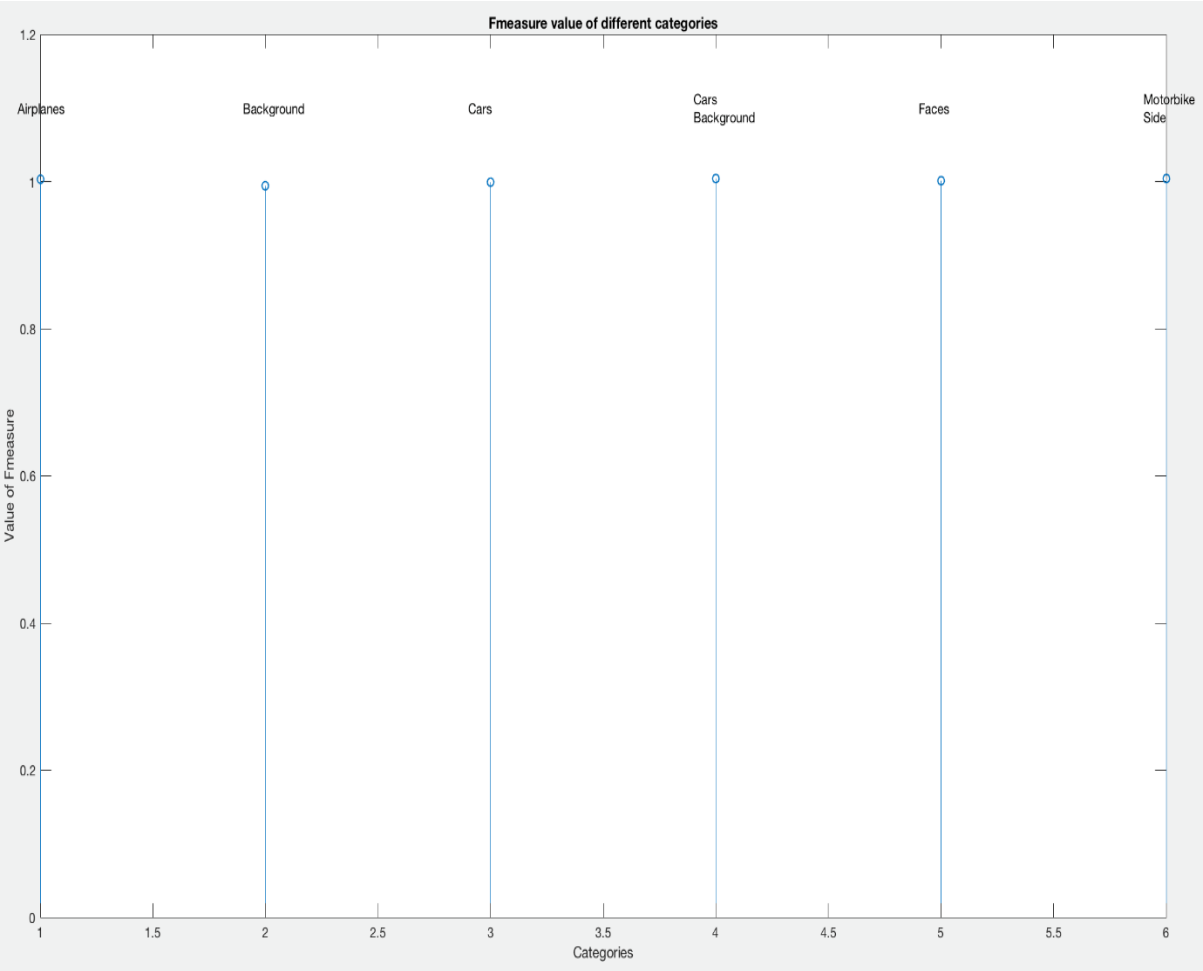

Fig 7.24 Fmeasure value of different categories

CONCLUSION

Through the implementation of this particular project I have been able to understand the various standard algorithms used in computer vision. In the modern era of technology in which everyone is dependent upon some kind or another, to use and implement computer vision algorithms is very important.

With the help of this project, I learned about many principles that led to the finishing of this project. First and foremost was the domain of Feature Detection. Under this, there was a clear distinction between the four primarily used feature detectors namely SURF, MSER, FAST and Harris Detector. Through extensive research and quantitative analysis, I was able to single out the most efficient and computationally fastest feature detector that was the right choice for this particular project. Next part of the project included the use of region segmentation for which I used K- Means Clustering algorithm. K-Means Clustering is industry stand algorithm to segment part of images and used the desirable on.

Furthermore, the advanced stages of the project demanded the implementation of Machine Leaning algorithm upon which today's computational giants stand. Getting a chance to use Support Vector Machine resulted in me obtaining a glimpse of what happens behind the scenes of most of the machine learning software. Last but not the least, the statistical analysis of the whole project presented a quantitative view of the performance and accuracy of the whole project using a total of four different measures namely Average Accuracy, Precision, Recall and Fmeasure.

REFERENCES

[1] T. Shukla, N. Mishra and S. Sharma, "Automatic Image Annotation using SURF Features", Proceedings of the International Journal of Computer Applications (0975 – 8887), Volume 68– No.4, April 2013.

[2] J. Li and J. Z. Wang, "Real-time Computerized Annotation of Pictures", Proceedings of the ACM Multimedia Conference, pp. 911-920, ACM, Santa Barbara, CA, October 2006.

[3] R. Datta, J. Li, and J. Z. Wang, "Content-based image retrieval - Approaches and trends of the new age," In proc. Int. Workshop on Multimedia Information Retrieval, pp. 253–262, 2005.

[4] J. Li and J. Z. Wang, "Automatic Linguistic Indexing of Pictures by a Statistical Modeling Approach", IEEE Trans. Pattern Analysis and Machine Intelligence, pp. 1075–1088, 2003.

[5] M. Oujaoura, B. Minaoui, M. Fakir, "A semantic Approach for Automatic Image Annotation", Proceeding of Intelligent Systems: Theories and Applications (SITA), 2013 8th International Conference, Rabat, Morocco, 8-9 May 2013, pp. 1-8 2013.

[6] F. Ali, S. Ullah Khan, M. Zarrar Mahmudi, R. Ullah, " A Comparison of FAST, SURF, Eigen, Harris, and MSER Features", Proceedings of International Journal of Computer Engineering and Information Technology, vol.8, no.6, pp.100-105, June 2016.

[7] P. M. Panchal, S. R. Panchal, S. K. Shah, "A Comparison of SIFT and SURF", Proceeding of International Journal of Innovative Research in Computer and Communication Engineering Vol. 1, Issue 2, April 2013.

[8] L. Teng, M. Tao., I. Kweon, "Contextual Bag-of-Words for Visual Categorization", Proceedings of IEEE Transactions on Circuit and Systems for Video Technology, Vol. 21, No. 4, April 2011.

[9]Lena_Coloured.jpg, "https://www.cosy.sbg.ac.at/~pmeerw/Watermarking/lena.html", accessed on 15 September, 2017.

[10]Kitchen.jpg, "http://www.antiquesl.com/latest-kitchen-design-trends-2016/beautiful-wall-tile-backsplash-for-american-kitchen-design-with-big-drum-pendant-light/", accessed on 20th September 2017.

[11] Hestain.png, "http://images.slideplayer.com/39/10999336/slides/slide_16.jpg", accessed on 7th November, 2017.

CURRICULUM VITAE (CV)

Eshan Gaur

Senior Year B. Tech Student, Electronics & Communication Engineering

Academic Details

Year	Degree	Institute	CGPA/Percentage
2014-present	B.Tech in Electronics & Communication	Jaypee Institute of Information Technology, Noida	CGPA=7.7/10 (up to 7th semester)
2014	Class XII CBSE	D.A.V. Public School, Shrestha Vihar	90.1%
2012	Class X CBSE	D.A.V. Public School, Shrestha Vihar	95%

Work Experience

June 19th ,2017 – July 29th ,2017

Summer Training Intern | Snapdeal | New Delhi

Projects

- Trend Visualizer using Raspberry Pi
  A web application for the real-time end-to-end analysis of selected Internet trends where the trend can be whatever the people post online.
- Automatic Image Annotation
  Aim of the project was to automatically annotate the image database using feature detection and SVM.

Technical Skills

C++, Python, MATLAB, C, Java, JavaScript