International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 08 Issue: 05 | May 2021 www.irjet.net p-ISSN: 2395-0072

An Approach Towards 3D Scene Reconstruction from an Uncalibrated Monocular Video Preceeded by an Exhaustive Study of Preprocessing of Images and Feature Detection Algorithms

Mouli Laha

National Informatics Centre, MietY, Govt of India

Abstract - *3D reconstruction of a scene is not only an emerging but also a challenging area of research work. In this work the novel approach is to develop a 3D reconstruction of the scene in a video sequence taken from a single moving uncalibrated camera. The first step of the process is to divide the video into number of frames and to detect and extract feature points. Then the requirement is to find matching correspondences within the adjacent frames, after which the frames can be stitched. In this connection, feature detection, extraction and matching have proved to be a very useful area in the study of image processing in recent times. It occupies a major section of research in the computer vision domain. But feature detection faces the problem of occlusion, view point dependence and scale. Several algorithms have been proposed till date for the feature detection purpose. Not only feature detecting proficiency, but a matching procedure has also been incorporated in which percentage of the match is calculated. Along with the matching proficiency, the speed of each algorithm is also compared by computing the processing time of the algorithm. Finally, a suitable and efficient algorithm is thus chosen for the feature detection purpose in 3D reconstruction process.*

Key words: 3D reconstruction, feature detection, extraction, matching, image processing, computer vision.

1. INTRODUCTION

The aim is to design a 3D structure from an uncalibrated video sequence. Since depth estimation is required, therefore stereo image could have eased the process. But since this work deals with a video sequence taken by a single uncalibrated moving camera, thus the requirement is to stitch the image frames from the video sequence.

Stitching will require a process of feature detection, extraction and finding matching correspondences within adjacent image frames. The detected features are basically keypoints which are capable of recognizing specific entities within an environment. This process is termed as feature detection. But the process of feature detection must not be carried out unless an image has been preprocessed. Basically, preprocessing refers to operating over the image at the lowest level of abstraction.

Image processing consists of various methods and tools which has been explained in [1]. In this work, noise removal operation has been considered in the preprocessing part. When an image is captured, it generally consists of various noises. Few noises have been explicitly incorporated within an image and the number of feature points detected by the feature detection algorithms has been analyzed. The noises considered in this case have been explained below in brief.

Gaussian Noise [2]: It is also termed as electronic noise since it arises in detectors or amplifiers. Gaussian noise is basically the sensor noise. It disturbs the gray level in a digital image. Gaussian noise is represented by the probability density function of the gray level. The PDF is represented in (1).

$$P(g) = \sqrt{\frac{1}{2\pi\sigma^2}} e^{\frac{-(g-\mu)^2}{2\sigma^2}} \qquad (1)$$

Where μ is the mean and σ is the variance. A Gaussian noise PDF curve with zero mean, 0.1 variance and 256 gray levels have been shown in Fig. 1.

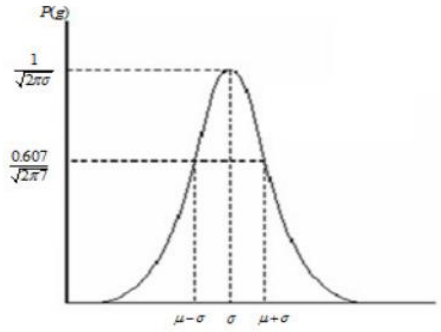


Fig-1: PDF of Gaussian noise.

- Poisson Noise [2]: This type of noise is also termed as photon noise or shot noise. It is a type of electronic noise which obeys Poisson's distribution. The presence of this noise is due to the statistical nature of electromagnetic waves such as x-rays, gamma rays and visible lights. The sources of x-ray and gamma ray emit number of photons per unit time. These sources have random fluctuation of photons which results in spatial

and temporal randomness in the gathered image. The Poisson distribution is represented in (2).

$$p(k\ events\ in\ interval) = \frac{\lambda^k e^{-\lambda}}{k!} \quad (2)$$

- Salt and Pepper Noise [2]: This kind of noise is basically visible as the sparse occurrence of black and white pixels. It is also known as impulse valued noise. Generally, this type of noise is found during data transmission and is also termed as data drop noise. In this noise type, an image pixel value is changed to either maximum or minimum values i.e. either '255' or '0' respectively as shown in Fig. 2. In salt and pepper noise, moderately bright pixel values are present in dark region and vice versa. Probability Density Function of the same has been shown in Fig. 3.

Fig-2: Central pixel value corrupted by salt and pepper noise

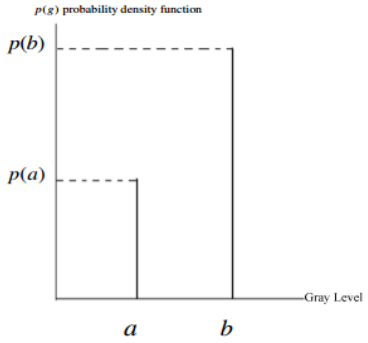


Fig-3: PDF of Salt & Pepper Noise

Where g denotes the gray level and

$$p(g) = \begin{cases} p(a) for\ g = a \\ p(b) for\ g = b \\ 0\ otherwise \end{cases}$$

and the PDF shows zero mean and 0.05 variance.

- Speckle Noise [2]: This kind of noise is basically multiplicative noise. This type of noise degrades the quality of radar, medical ultrasound, and optical coherence images. Probability density function of speckle noise follows gamma distribution and is represented in (3).

$$p(g) = \frac{g^{a-1} e^{\frac{-g}{a}}}{(a-1)! a^a} \quad (3)$$

After detecting the key points within the noisy images, various noise removal filters have been applied to detect the key points within filtered images.

Filtration is basically an image enhancement technique. Filtration leads to two kinds of resultant images: Smoothened image and Sharpened image; based on whether we have used low pass or high pass filters respectively.

A. Low Pass Filter [3]: This kind of filters allows only low frequency signals to pass than a mentioned cut-off frequency and also attenuates high frequency signals, higher than the cut-off frequency. Low pass filters result in smoothened or slightly blurred image. Few standard forms of the low pass filters we have used for our experimental study are adaptive, average/mean, Gaussian and median filter.

B. High Pass Filter [3]: This filter attenuates low frequency signals; lower than a specified cut-off signal and passes only higher frequency signal than the cut-off signal. High pass filtering leads to sharpening of the image. The standard form of high pass filter which we have used is the Gaussian high pass filter.

The filters considered in this study have been explained below.

- Adaptive Filter [4]: This type of filter is a linear digital filter with a transfer function with variable parameters and an optimized means to adjust those parameters. The working principle of Wiener filter, which is an adaptive filter, is based on a statistical approach [5], shown in Fig. 4.

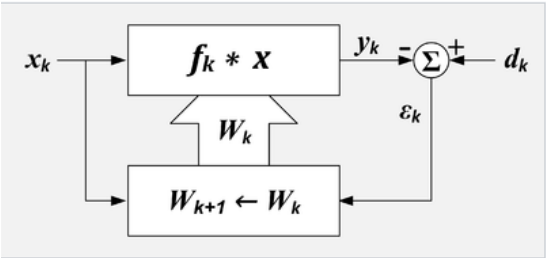


Fig-4: Adaptive filter

In Fig. 4., k = sample number, x = reference input, X = set of recent values of x, d = desired input, W = set of filter coefficients, ε = error output, f = filter impulse response, * = convolution, Σ = summation, upper box is the linear filter and lower box is the adaption algorithm.

Fig. 5. is an adaptive linear combiner showing the combiner and the adaption process.

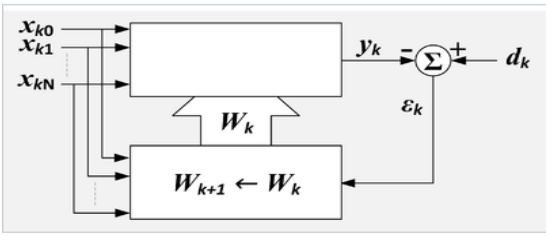


Fig-5: Adaptation process

- Where, k = sample number, n=input variable index, x = reference inputs, d = desired input, W = set of filter coefficients, ε = error output, Σ = summation, upper box is the linear combiner and lower box is the adaption algorithm.
- Average Filter [4]: Also called mean filter and it uses linear filtering to remove noises from an image. In mean filtering, each image pixel value is replaced with the mean value of its neighboring pixels, including itself. Average or mean filter is basically thought of as a convolution filter. Mean filter is defined by (4).
- $Mean\_filter_{(x_1 \dots x_N)} = \frac{1}{N}\sum_{i=1}^{N} x_i$ (4)
- Where $(x_1 \dots x_N)$ is the image pixel range.
- Gaussian Filter [6]: The impulse response of a Gaussian filter is a Gaussian function. A Gaussian filter, shown in Fig. 6. is parameterized by its mean μ and variance σ^2 as given in (5).
- $G(\mu, \sigma^2, t) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{\frac{-(t-\mu)^2}{2\sigma^2}}$ (5)
- Where t runs between [-∞, ∞].

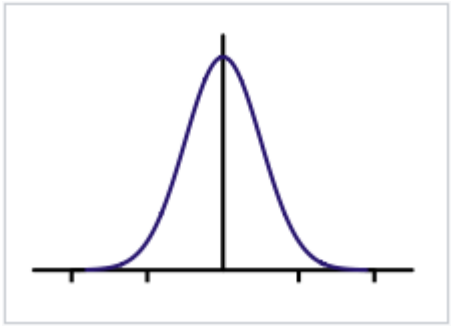


Fig-6: Shape of the impulse response of a Gaussian filter

- Median Filter [4]: This kind of filter is a digital filter which is non-linear. Median filter works by replacing each signal entry with the median of its neighboring entries. For odd number of entries, the median is easy to define. It is the middle value of the sorted neighboring entry values. For even number of entries, more than one median value exists. One major advantage of median filter being a non-linear filter is that median filter can eliminate consequence of extremely large valued input noise.

124 126 127
120 150 125
115 119 123

From the above matrix, sorted neighboring values are 115, 119, 120, 123, 124, 125, 126, 127 and 150. The median value is 124. Therefore, the filtered matrix becomes

124 126 127
120 124 125
115 119 123

Now after preprocessing the image by filtering and removing noise from the captured frame, next comes the feature detection phase. Basically, any recognizable key point in an image constitute a feature in that image.

Features can be categorized into various types as listed below:

- Geometric features: Mainly points, lines, plane or quadratic surface qualify as geometric features. They provide considerable constraints in the interpretation of sensor measurements. Point data sets can form a geometric primitive such as corners. Also, one of the significant geometric feature modelling is points of maximum curvature.
- Natural features: Outdoor environment cannot be modelled easily using geometric features. Thus, other modelling approaches have been proposed. One such approach is grid-based modelling. Unlike geometric feature modelling methods, grid models do not provide any constraint on the interpretation of measurement information.
- Generalized features: Basically, these are non-points features. Any sensor detectable entity fundamentally qualifies as generalized feature. Blob detection proves helpful in recognizing general features since blob feature model can accommodate a large number of feature characteristics.

Detailed discussion over feature types can be found in [7].

Features now need to be extracted after being detected. But feature extraction faces three major problems:

a. Occlusion: This refers to invisibility of some part of the terrain to the sensor due to shadowing or blockage of the view. Occlusion can lead to noisy and inappropriate results.

b. View point dependence: According to the viewing location or perspective of the sensor, spatial appearance of the object gets changed. But few general characteristics of an object do not change with the changing viewpoint. Points of maximum curvature and corners falls under such category. Many algorithms have been developed for point of curvature and corner detection such as Harris, Fast, and SUSAN etc.
c. Scale: Geometry of natural features at times depends on the resolution or scale of the objects. Large scale provides a global view of the shape of the object whereas finer scale provides detailed features of the overall shape. In finer scale level, the features decompose to multiple sub features. Finding a feature correlation between different viewpoints is naturally a function of scale. Multiple resolution feature representations can be done by scale space filtering. Basically, this is done by smoothening the image or data successively by Gaussian kernel. SIFT and SURF algorithms work by this principle of Gaussian smoothing.

These problems have been discussed in detail in [7].

From the above-mentioned problems faced during feature extraction, the problem of occlusion cannot be taken care of. But the remaining two problems can be solved using various algorithms of feature detection.

View point dependency can be handled if points of curvature or corners are considered for detecting features. Harris and FAST algorithm have been considered for detecting features using Laplacian functions.

Harris and FAST, however, are not scale invariant. Thus, the problem of scale can be handled by Gaussian kernel smoothing which is the basis of SIFT and SURF algorithm.

In "Review of 3D reconstruction from video sequences" [18], it can be seen that for feature detection and matching, few algorithms have been suggested, but due to unavailability of an exhaustive evaluation of those algorithms, no particular algorithm could be suggested with certainty. In this paper the target is to provide such a kind of exhaustive evaluation to find out which feature detector serves the purpose of detection and matching with utmost efficiency.

The above mentioned four algorithms, i.e. SIFT, SURF, FAST and Harris have been considered to compare their efficiency in detecting and matching feature points. The results are then analyzed and the most efficient algorithm is considered further for finding correspondences between the frames, segregated from the video sequence taken from an uncalibrated single moving camera.

The working principles of the algorithms are explained below in brief.

A. Scale Invariant Feature Transform (SIFT) [8]: It is a local feature detector and descriptor. SIFT mainly detect the interest points or keypoints in an image. The algorithm works by detecting the keypoints in an image o from a set of training examples and storing them in a database. Later when a sample image is provided, the SIFT algorithm can easily detect the features in that image.

SIFT basically consists of four steps [9]:

1. Scale-space extrema detection: this consists of computation searches over all scales and image locations. Difference-of-Gaussian function is used to identify potential interest points that are invariant to scale and orientation.

   Scale space of an image is defined as a function, say L(x, y, σ), that is produced from the convolution of a variable-scale Gaussian, G(x, y, σ), with an input image, I(x, y) shown in (6) and (7).

$$L(x, y, \sigma) = G(x, y, \sigma) * I(x, y) \quad (6)$$

   Where * is the convolution operation in x and y, and

$$G(x, y, \sigma) = \frac{1}{2\pi\sigma^2} e^{-(x^2+y^2)/2\sigma^2} \quad (7)$$

   But stable keypoints can be efficiently detected if difference-of-Gaussian function is convolved with the image as given in (8).

$$D(x, y, \sigma) = (G(x, y, k\sigma) - G(x, y, \sigma)) * I(x, y)$$
$$= L(x, y, k\sigma) - L(x, y, \sigma) \quad (8)$$

   The input image is convolved with Gaussians to produce the set of scale space images shown on the left. Neighboring Gaussian images are subtracted to produce the difference-of-Gaussian images on the right. This process takes place for first sample, after which the Gaussian image is downsampled by a factor of two and the above process is again repeated. The pictorial representation is shown in Fig. 7.

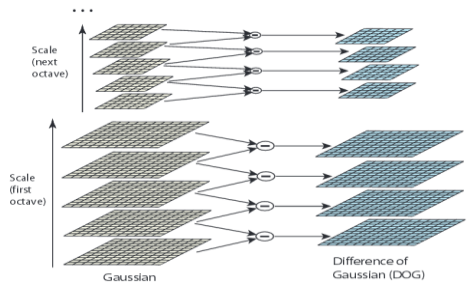


Fig-7: The convolved images are shown in left and the difference-of-Gaussian images have been shown in right.

2. Keypoint localization: The scale and location are determined by imposing a detailed model on each candidate location. Keypoints are detected base on their stability.

3. Orientation assignment: Each keypoint location is assigned one or more orientations based on local image gradient location. Therefore, the image data become invariant to these transformations.
Let an image sample be L(x, y), at the chosen scale, the gradient magnitude be m(x, y) and orientation be θ(x, y). The gradient an orientation is precompiled using pixel differences as given in (9) and (10).

$$m(x,y) = \sqrt{(L(x+1,y) - L(x-1,y))^2 + (L(x,y+1) - L} \quad (9)$$

$$\theta(x,y) = tan^{-1}\,(L(x,y+1) - L(x,y-1))/(L(x+1,y) - 1 \quad (10)$$

4. Keypoint descriptor: The magnitude of the local image gradient and the orientation of image points within the image region constitute the keypoint descriptor. These points are then weighted by a Gaussian window, after which these samples are assembled into an orientation histogram as shown in Fig. 8.

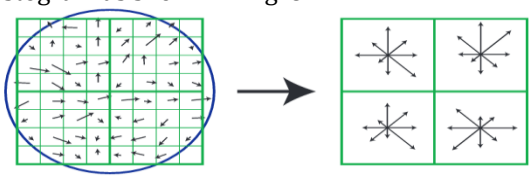


Fig-8: Orientation of image points through Gaussian window is shown on left and a 4*4 orientation histogram is shown on right.

B. Speeded Up Robust Features (SURF) [10]: SURF is a feature detector and descriptor that can be used for object detection and recognition. SURF uses an integer approximation of the determinant of Hessian based blob detector to detect the interest points. SURF uses square-shaped filters as an approximation of Gaussian smoothing, also known as Gaussian blur which is basically convolution of the image with a Gaussian function.

Square filtering is fast if integral image is used, given in (11).

$$S(x,y) = \sum_{i=0}^{x}\sum_{j=0}^{y} I(i,j) \quad (11)$$

Blob detector based on the Hessian matrix to find points of interest.

Suppose f: $\mathbb{R}^n \rightarrow \mathbb{R}$ is a function taking x ϵ $\mathbb{R}^n$ as input a vector and outputting a scalar f(x) ϵ $\mathbb{R}$. If all second partial derivatives of f exist and are continuous over the domain of the function, then the Hessian matrix H of f is a square n×n matrix and is represented as given in (12).

$$H_{i,j} = \frac{\partial^2 f}{\partial x_i \partial x_j} \quad (12)$$

Given a point p=(x, y) in an image I, the Hessian matrix H (p, σ) at point p and scale σ, is shown in (13).

$$H(p,\sigma) = \begin{pmatrix} L_{xx}(p,\sigma) & L_{xy}(p,\sigma) \\ L_{xy}(p,\sigma) & L_{yy}(p,\sigma) \end{pmatrix} \quad (13)$$

Where $L_{xx}(p,\sigma), L_{xy}(p,\sigma)$ & $L_{yy}(p,\sigma)$ are the second-order derivatives of the grayscale image.
The approximated convolution for arbitrarily sized kernel can be calculated, using (14), utilizing the integral image.

$$Det(H_{approx}) = D_{xx}D_{xy} - (wD_{xy})^2 \quad (14)$$

Where the approximated and discrete kernels are referred to as D_{yy} for $L_{yy}(p,\sigma)$ and D_{xy} for $L_{xy}(p,\sigma)$. The working has been shown in Fig. 9, Fig. 10. and Fig. 11.

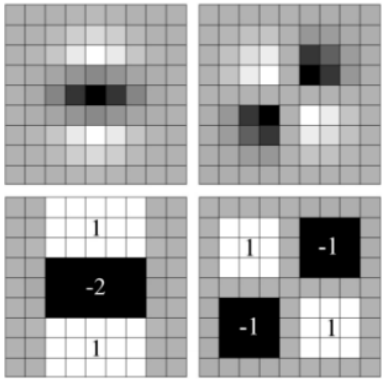


Fig- 9: $L_{yy}(p, \sigma)$ & $L_{xy}(p, \sigma)$ shown at the top left and right respectively and the corresponding approximated and discrete kernels D_{yy} & D_{xy} shown at bottom left and right respectively.

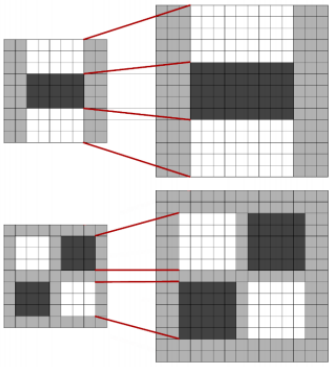


Fig-10: The size of the kernels is increased keeping the lobes properly scaled.

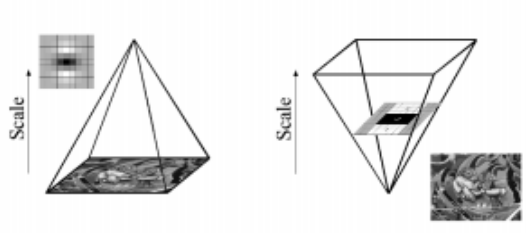


Fig-11: SIFT downscales the image as shown in left whereas SURF use larger filters as shown in right.

C. Features from Accelerated Segment Test (FAST) [11]: The algorithm was originally proposed by Rosten and Drummond [12]. Basically FAST algorithm is used for detecting corner points in an image. The algorithm works as follows:

- A pixel p in an image is selected. Let the intensity of the pixel be I_p. Now whether this pixel will be detected as an interest point or not, that needs to be found out.
- A threshold intensity value is set; say T.
- A circle of 16 pixels is the considered which surrounds the pixel p.
- For the pixel p to be detected as an interest point, N contiguous pixels need to be either above or below I_p out of those 16 pixels.
- Compare the intensity of pixels 1, 5, 9 and 13 of the circle with I_p to make the algorithm fast.
- p is not an interest corner point if at least three of the four pixels, I_1 I_5 I_9 I_{13} are not above or below I_p +T.
- Else if at least three are above or below I_p +T, check for all 16 pixels whether 12 contiguous pixels fall in the criterion.
- Repeat the above procedure for all pixels in the image.

The detection of the corner points has been shown in Fig. 12.

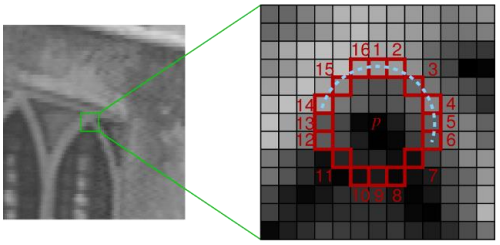


Fig-12: Corner point detection in FAST.

D. Harris operator [13][14]: It is a corner detection method which is used for detecting objects. Corner is intersection of two edges i.e. corner is a point with two different and dominant edge directions within the local neighborhood of the point. Initially corner detection used to be carried out using the concept of correlation which was computationally expensive and suboptimal. But with the invention of Harris operator, proposed by Harris and Stephens, corner detection became easy.

A two-dimensional grayscale image is considered for detecting corner points, say I. Considering a point (x, y) over the image and a small shift, say (Δx, Δy), the auto-correlation function is defined as in (15).

$$c(x,y) = \sum_W [I(x_i, y_i) - I(x_i + \Delta x, y_i + \Delta y)]^2 \quad (15)$$

Where W is a Gaussian window and (x_i, y_i) constitute the points within that window.

The shifted image is approximated using Taylor expansion is given in (16).

$$I(x_i + \Delta x, y_i + \Delta y) \approx I(x_i, y_i) + \left[I_x(x_i, y_i) I_y(x_i, y_i)\right]_{\Delta y}^{\Delta x}$$

(16)

Where I_x (.,.) and I_y (.,.) are the partial derivatives in x and y respectively.

From (15) and (16), we get (17).

$$c(x, y) = (\Delta x, \Delta y)\, C(x, y) \begin{matrix} \Delta x \\ \Delta y \end{matrix} \quad (17)$$

Where C(x, y) denotes the intensity structure of the local neighborhood.

Let λ_1 and λ2 denotes the eigenvalues of the matrix C(x, y). There are the following three cases which need to be considered:

- If λ1≈0 and λ2≈0, then this pixel (x, y) has no features of interest.
- If λ1≈0 and λ2 has some large positive value, then an edge is found.
- If λ1 and λ2 have large positive values, then a corner is found.

2. RELATED WORKS

Many approaches to 3D reconstruction have been done till date. This work is another such approach towards this field. Few works already done in this area have been mentioned below:

1. Frahm et al. [15] carried out extraction and matching of feature points have been using View Point Invariant patch, which is an extension of SIFT. Sparse reconstruction of the video sequence shows triangulated 3D points. Vocabulary tree approach has been used to find likely matches which are built by employing k-means clustering to find high dimensional features. Finally bundle adjustment is done and a dense model is generated.
2. Han et al. [16] have partitioned a video into multiple scenes, followed by sparse reconstruction, which starts by finding feature correspondences and then stitching. The final step is dense reconstruction. This paper provides a novel approach to acquiring dense depth map. In this case the static background is assumed to be decomposed into multiple parametric surfaces, which allows the 3D point cloud to be partitioned into multiple clusters according to their geometric properties.
3. Rodriguez et al. [17] took up a 3D scene reconstruction approach. Online calibration of camera has been done in this case. Secondly, a set of 3D points is provided along with projection matrices of the set of images and 3D to 2D point correlation. Mesh generation is then carried out. Finally, a 3D view is created using the Visire authoring tool.
4. Trung Kien et al. [18] presented a report on reconstructing a 3D scene which serves as a software prototype in "Crime scene investigation using hand-held cameras". The limits set for the report are that the video must consist of static scenes taken from uncalibrated camera of varying intrinsic camera parameters.

The report explained the steps involved in 3D reconstruction which can are as follows:

a) Take an input video.
b) Feature detection and matching.
c) Structure and motion recovery.
d) Stereo mapping.
e) Modelling.
f) 3D model reconstruction.

5. Repko et al. [19] considered two major issues namely, key frame selection and projective drift. In key frame selection process, the camera motion can be determined reliably by evaluating the feature correlation between three consecutive views. For projective drift issue, self-calibration approach has been proposed which is unresponsive to the projective drift. After taking care of the above-mentioned issues, stereo matching is carried out. Finally, a 3D model is textured.
6. Dmre et al. [20] addressed the problem of multi frame structure from motion for monocular video sequence in dynamic scenes. An algorithm has been proposed to solve this problem. Feature tracking and segmentation are done followed by prioritized sequential reconstruction.
7. Pollefeys et al. [21] dealt with automatic, geo-registered and real-time 3D reconstruction. Video of urban scenes have been used for the reconstruction purpose. GPS video data have been used. The processing modules consist of:

a) Reading the input data.
b) 2D tracking of the pertinent image features.

c) 3D tracking to estimate the camera poses.
d) Sparse scene analysis.
e) Stereo depth estimation.
f) Depth map fusion.
g) Model generation.

From the above works, it can be perceived that feature detection, extraction and matching is the initial and primary step for the 3D reconstruction process. This paper, therefore, serves the purpose of selecting an appropriate feature detector for carrying out the rest of the process.

Also, many kinds of comparative evaluation have also been done as mentioned below:

1. Juan et al. [22] performed a comparative study on the efficiency of SIFT, PCA-SIFT and SURF algorithms. Evaluation has been done on the basis of several alterations namely rotation, blur, illumination changes and affine transformations. Tabular and graphical display of results has been provided.
2. Mikolajczyk et al. [23] have done an elaborated comparative analysis of various local descriptors namely SIFT, PCA-SIFT, GLOH, shape context, spin images, cross correlation, steerable filters, differential invariants, complex filters and moments. The study consists of matching precision of the algorithms based on various transformations like rotation, blur, scale changes, illumination changes and JPEG compression.
3. Maini et al. [24] analysed various image edge detectors. Basically the comparative study consists of the proficiency of the existing edge detectors. The algorithms which have been considered for evaluation in this study are Sobet, Prewitt, Robert, Canny, Laplacian and Laplacian-of-Gaussian.
4. Ghosh et al. [25] considered the existing feature detection algorithms namely Harris, SURF, FAST and FREAK for a comparative analysis. Tabular and graphical analyses have been shown in their paper.
5. Saipullah et al. [26] performed another comparative analysis of real time feature detection. The object detection has been performed for an embedded system such as an Android smart phone. Object detection capability for FAST, SIFT, SURF, ORB, MSER. GFTT and STAR algorithms has been evaluated in this case.

3. FEATURE DETECTION AND EXTRACTION PROCESS

The experiments have been carried out in MATLAB. An input image has been used to carry out the experiments.
At first various noises are incorporated explicitly in the input image as shown in Fig. 13.

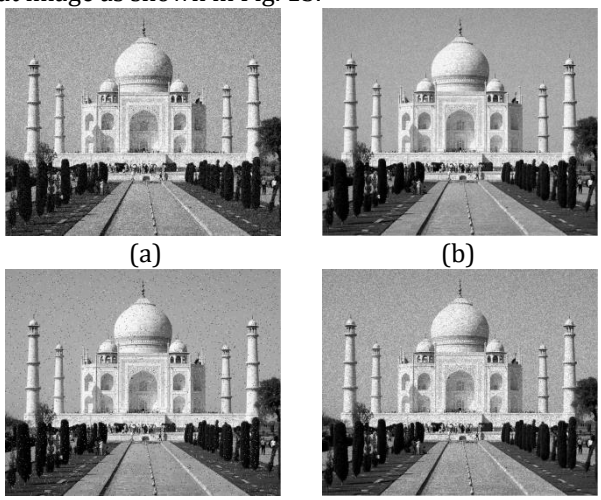


Fig- 13: (a) Gaussian noise. (b) Poisson noise. (c) Salt & Pepper noise. (d) Speckle noise.

Features are then detected within the above images using the feature detection algorithms. Detected features can be shown in Fig. 14 (for SIFT), Fig. 15 (for SURF), Fig. 16 (for FAST) and Fig. 17 (for Harris).

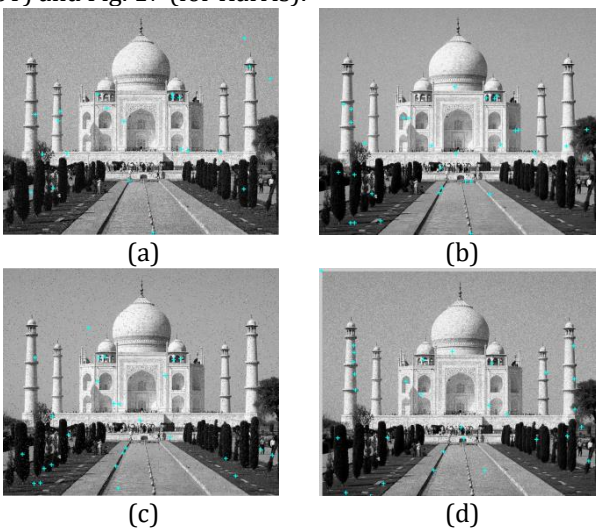
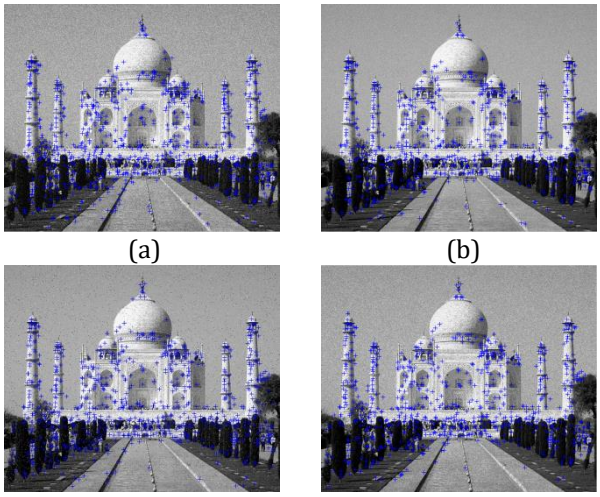


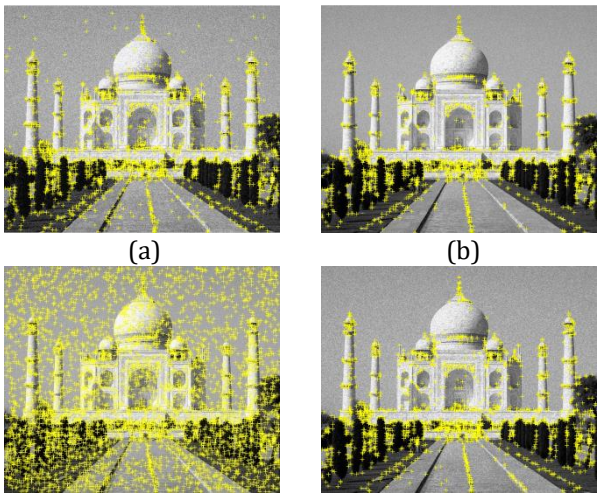
Fig-14: Features detected by SIFT detector in case of images with (a) Gaussian noise (b) Poisson noise (c) Salt & Pepper noise and (d) Speckle noise



(a) (b)

(c) (d)

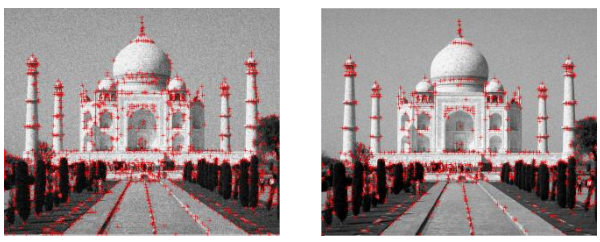
Fig-15: Features detected by SURF detector in case of images with (a) Gaussian noise (b)Poisson noise (c)Salt & Pepper noise and (d) Speckle noise



(a) (b)

(c) (d)

Fig-16: Features detected by FAST detector in case of images with (a) Gaussian noise (b)Poisson noise (c)Salt & Pepper noise and (d) Speckle noise

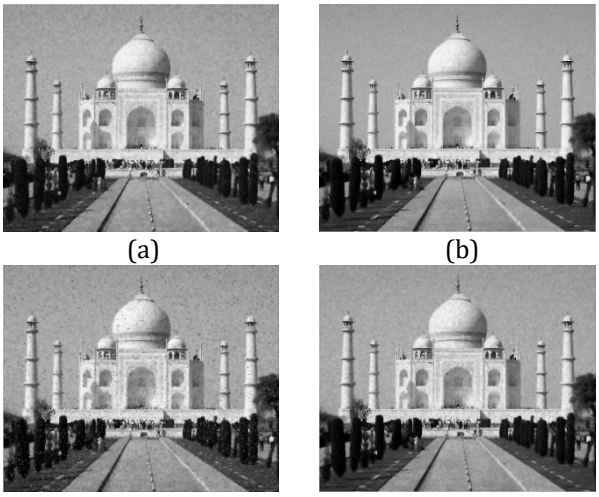


(a) (b)

(c) (d)

Fig-17: Features detected by Harris detector in case of images with (a) Gaussian noise (b)Poisson noise (c)Salt & Pepper noise and (d) Speckle noise

After incorporating various noises, various filters have been applied to remove the noise from the images. The images after noise removal by low pass filters have been shown in Fig. 18 (for Adaptive Filter), Fig. 19 (for Average Filter), Fig. 20 (for Gaussian Filter) and Fig. 21 (for Median Filter).



(a) (b)

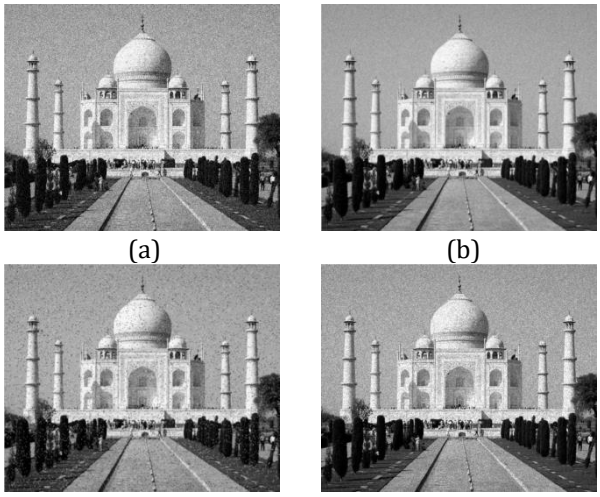
(c) (d)

Fig-18: Noise removal process carried out by Adaptive filter over images with (a) Gaussian noise (b)Poisson noise (c)Salt & Pepper noise and (d) Speckle noise

(a) (b)

(c) (d)

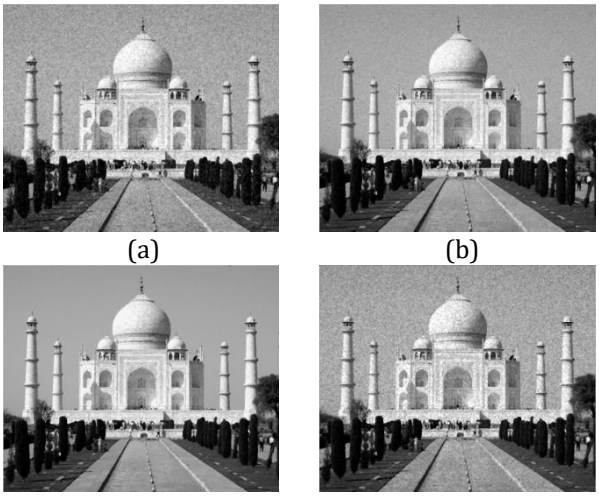
Fig-19: Noise removal process carried out by Average/Mean filter over images with (a) Gaussian noise (b)Poisson noise (c)Salt & Pepper noise and (d) Speckle noise



(a) (b)

(c) (d)

Fig-20: Noise removal process carried out by Gaussian filter over images with (a) Gaussian noise (b)Poisson noise (c)Salt & Pepper noise and (d) Speckle noise

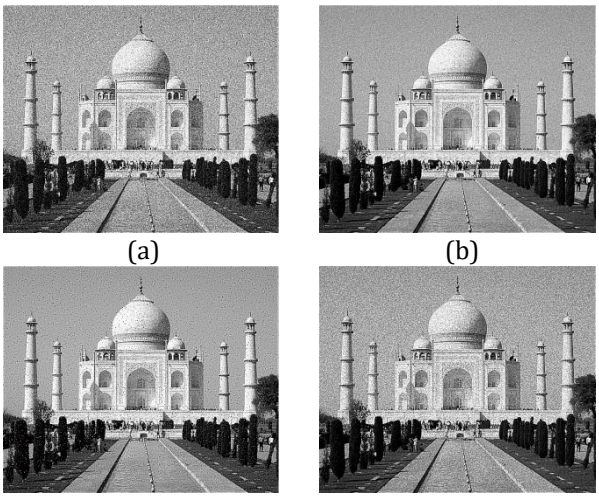


(a) (b)

(c) (d)

Fig-21: Noise removal process carried out by Median filter over images with (a) Gaussian noise (b)Poisson noise (c)Salt & Pepper noise and (d) Speckle noise

Apart from low pass filters, the results have also been analyzed with high pass Gaussian filter which has been shown in Fig. 22.



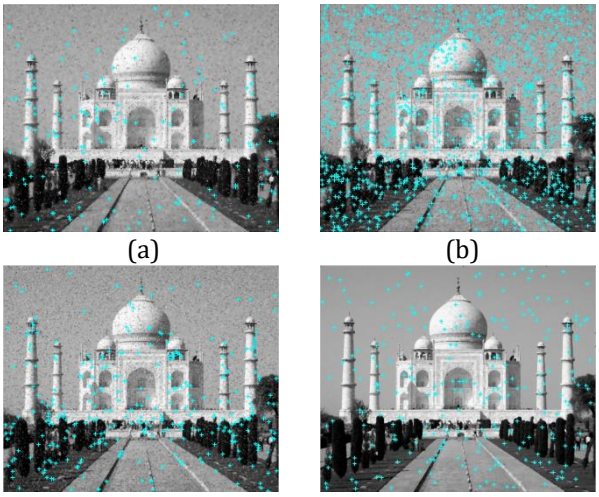
(a) (b)

(c) (d)

Fig-22: Noise removal process carried out by high pass Gaussian filter over images with (a) Gaussian noise (b)Poisson noise (c)Salt & Pepper noise and (d) Speckle noise

It has been observed from the above images that low pass filter is basically responsible for smoothening an image, as a result of which the image gets blurred. On the other hand, high pass filtering results in sharpened image, making the features of an image more prominent.

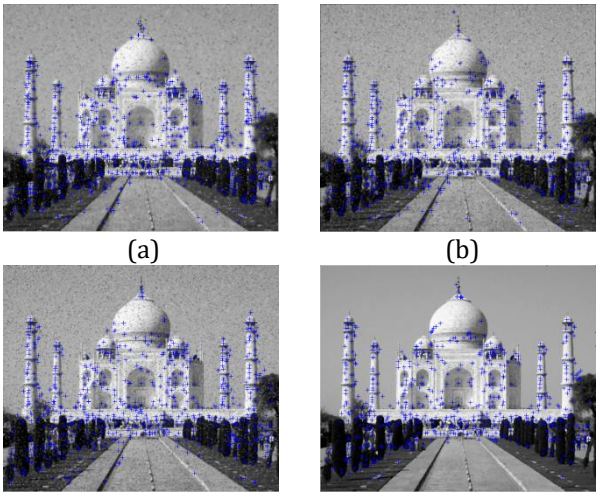
After filtration, feature detection algorithms are applied over the filtered images. The detected features by the feature detection algorithms can be shown as below in Fig. 23 (for SIFT), Fig. 24 (for SURF), Fig. 25 (for FAST) and Fig. 26 (for Harris).



(a) (b)

(c) (d)

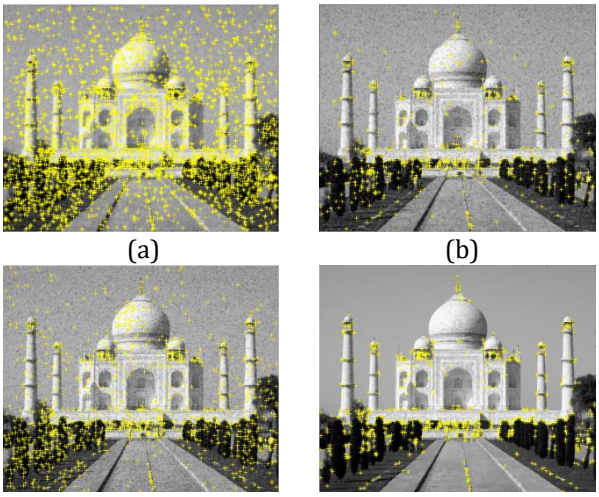
Fig-23: Feature detected by SIFT detector on images filtered by (a) Adaptive filter (b) Average filter (c) Gaussian filter and (d) Median filter.



(a) (b)

(c) (d)

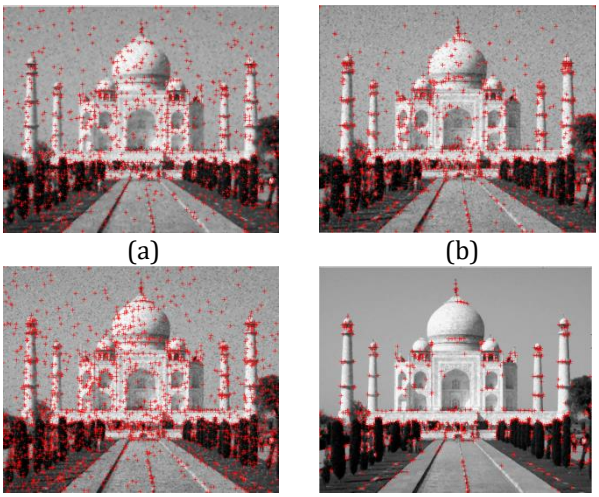
Fig-24: Feature detected by SURF detector on images filtered by (a) Adaptive filter (b) Average filter (c) Gaussian filter and (d) Median filter.



(a) (b)

(c) (d)

Fig-25: Feature detected by FAST detector on images filtered by (a) Adaptive filter (b) Average filter (c) Gaussian filter and (d) Median filter.

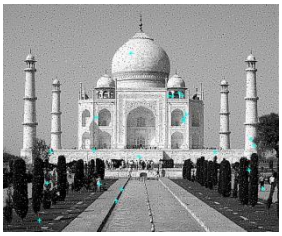


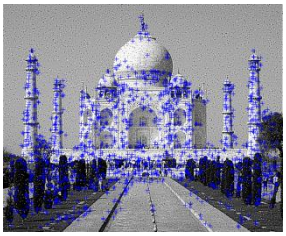
(a) (b)

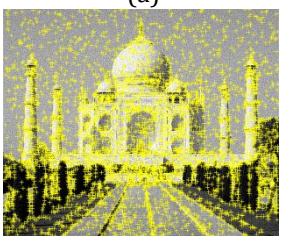
(c) (d)

Fig-26: Feature detected by Harris detector on images filtered by (a) Adaptive filter (b) Average filter (c) Gaussian filter and (d) Median filter.

The above figures showed us the result of feature detection over low pass filtering of images. Features detected by the algorithms over high pass filtered image have been shown in Fig. 27


(a)


(b)


(c)

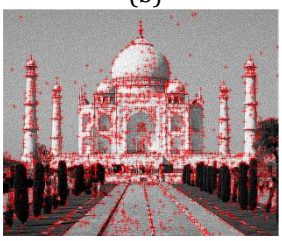

(d)

Fig-27: Feature detected on high pass filtered images by (a) SIFT detector (b) SURF detector (c) FAST detector and (d) Harris detector.

After preprocessing of the images and detecting features from filtered images, the efficiency of the feature detection algorithms is analyzed in case if the image is morphed. Comparisons have been done with the image after various kinds of transformations. The modifications that have been done are explained below:

- Crop: The sample image in Fig. 28. is compared with a cropped version of the same image, shown in Fig. 29. At first the features are detected from the cropped image and then the matching is done.
- Rotation: The sample image is rotated 90˚ (in Fig. 30.) and its features are detected using the mentioned algorithms. The features are then matched with the original image and the match percentage is calculated.
- Blur: the original image is blurred (in Fig. 31.) and then is used for the feature detection and extraction purposes. The detected features are matched with the detected features of the original image and the percentage of match is evaluated.


Fig-28: Original image


Fig-29: Cropped image

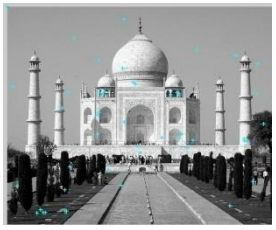

Fig-30: Rotated image


Fig-31: Blurred image

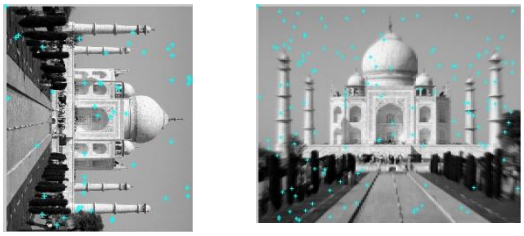
After detecting, extracting and matching the features of the original image with the altered versions of the image, another set of matching experiment has been performed. In this case, the original image is compared with a group of images, and the matching percentage is found out. The original image is then detected within the group of images (in Fig. 32.) by the number of matched features. The image within the set of images which corresponds to maximum matched features is identified to be the original image.


Fig-32: The set of images used for the matching experiment

Feature detected by the SIFT descriptor for the original as well as modified images are shown below in Fig. 33.


(a)

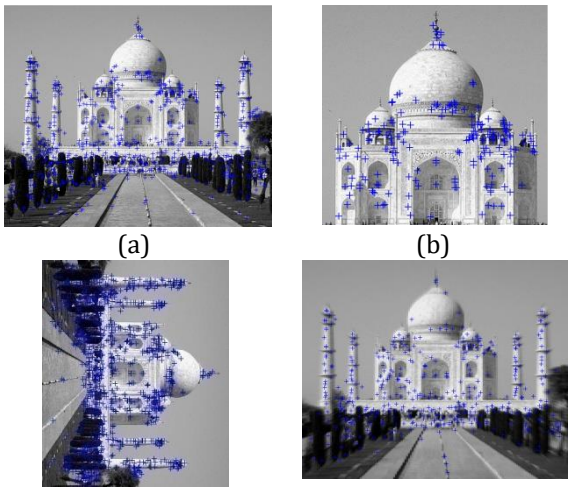

(b)



(c) (d)

Fig-33: Feature detected by SIFT detector in (a) original image (b) cropped image (c) rotated image & (d) blurred image

Feature detected by the SURF detector has been shown in Fig. 34.



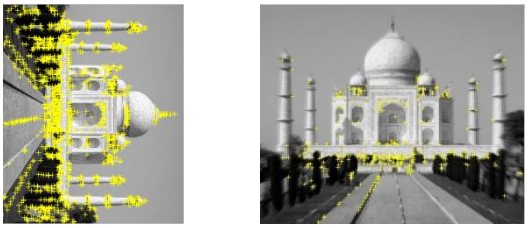
(a) (b)

(c) (d)

Fig-34: Feature detected by SURF detector in (a) original image (b) cropped image (c) rotated image & (d) blurred image

Similarly, FAST feature point detector also detected the key points from the images. The interest points detected are shown in Fig. 35.

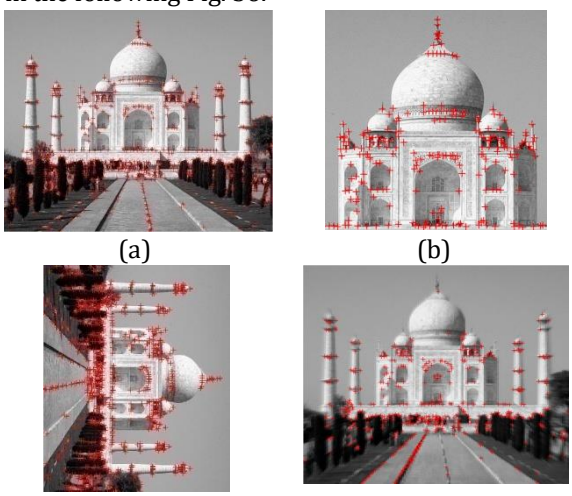
(a) (b)



(c) (d)

Fig-35: Feature detected by FAST detector in (a) original image (b) cropped image (c) rotated image & (d) blurred image

Finally, the Harris detector was applied over the images and the features detected by the Harris operator are shown in the following Fig. 36.

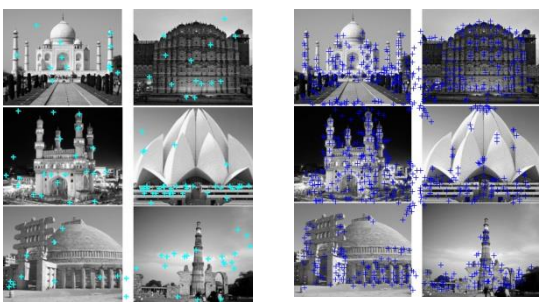


(a) (b)

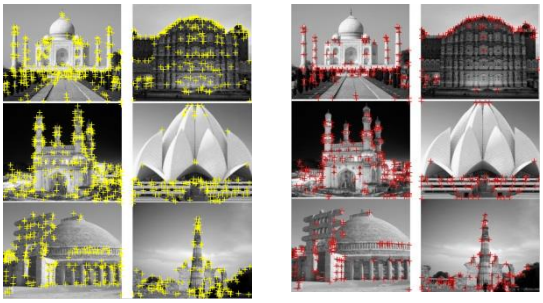
(c) (d)

Fig-36: Feature detected by Harris detector in (a) original image (b) cropped image (c) rotated image & (d) blurred image

Apart from detecting features in the above images, the four feature detectors also detected keypoints from a set of images, which have been used further for the matching process. The keypoints detected can be shown in the Fig. 37.



(a) (b)



(c) (d)

Fig-37: Feature detected by (a) SIFT detector (b) SURF detector (c) FAST detector & (d) Harris detector.

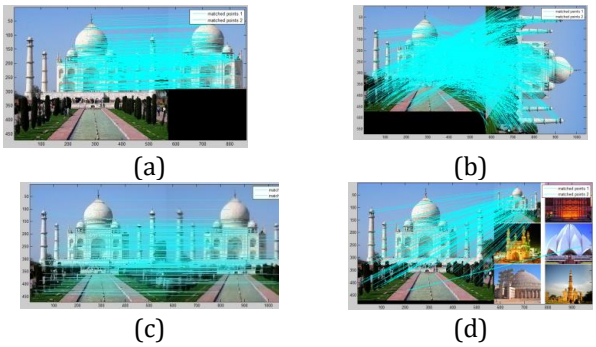
4. FEATURE MATCHING PROCESS

After extraction and detection of features from the images and set of images, the above-mentioned algorithms are applied for the matching process. In this process, keypoints are extracted from two sets of images. After extraction, the keypoints of one image is matched with the other image and the number of matches is found. The number of matches is then converted to percentage using the following relation:

$$Match\ \% = \frac{Number\ of\ matches\ found}{Number\ of\ keypoints\ in\ the\ original\ image}$$

The mapping of the matches after applying the algorithms can be shown by placing both the images side by side and keypoint from first image being matched to the corresponding keypoints in the second image. Few keypoints matched image pair is shown below.

Following Fig. 38. shows the matched pairs in which SIFT detector has been used for matching.

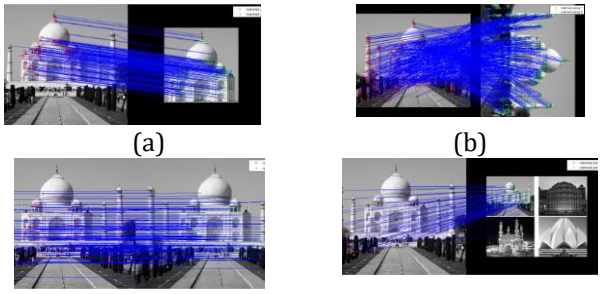


(a) (b)

(c) (d)

Fig-38: Feature matching of the original image done by SIFT detector with (a) the cropped image (b) rotated image (c) blurred image & (d) a set of images.

SURF detector carried out the matching process efficiently which is shown in Fig. 39.

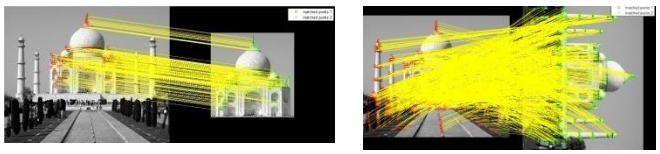


(a) (b)

(c) (d)

Fig-39: Feature matching of the original image done by SURF detector with (a) the cropped image (b) rotated image (c) blurred image & (d) a set of images.

After feature matching by SIFT and SURF detectors, FAST detector is then used for finding out matched keypoints between pairs of images. The matched keypoints is shown in Fig. 40.



(a) (b)

Fig-40: Feature matching of the original image done by FAST detector with (a) the cropped image (b) rotated image.

Finally, Harris feature detection operator is used to carry out the matching process as shown in Fig.41.

(a) (b)

Fig-41: Feature matching of the original image done by Harris detector with (a) the cropped image & (b) blurred image.

5. EXPERIMENTAL RESULTS

After completing the feature detection, extraction and matching process, the number of keypoints detected by each algorithm is calculated. Also, the number of matched features points is evaluated. The currently captured image is displayed first. Apart from the keypoint determination, the processing time for each algorithm is also calculated to ensure that the comparative study includes efficiency as well as speed of each algorithm.

An elaborated evaluation has been shown below in which number of keypoints detected as well as matched keypoints are calculated for the input image when it is matched with its modified versions.

Following are the data represented in tables regarding the number of feature point detection during the preprocessing phase, i.e. keypoints in noisy image as well as in the filtered images.

a) Feature detection and processing time taken in case of noisy images:

Table -1: Number of Keypoints Detected By Each Algorithm for Noisy Images

Algorithm	# of keypoints in image with Gaussian noise	# of keypoints in image with Poisson noise	# of keypoints in image with Salt & Pepper noise	# of keypoints in image with Speckle noise
SIFT	1607	1422	1550	1504
SURF	854	788	827	829
FAST	1847	1440	2101	1649
Harris	1167	807	1529	1144

The graphical representation is given in Chart -1.

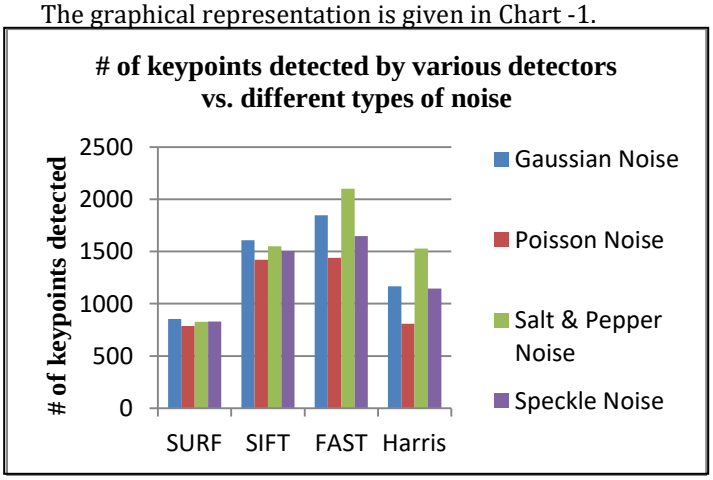


Chart -1: Graphical analysis of keypoint detection by each algorithm in case of noisy images.

Not only the number of keypoints, but also the processing time taken by each algorithm has been computed. The tabular representation has been shown below.

Table -2: Processing Time Taken By Each Algorithm For Feature Detection In Case Of Noisy Images

Algorithm	Image with Gaussian noise (in seconds)	Image with Poisson noise (in seconds)	Image with Salt & Pepper noise (in seconds)	Image with Speckle noise (in seconds)
SIFT	1.8612	1.8158	1.8451	1.8438
SURF	0.5994	0.5670	0.6173	0.5944
FAST	0.1827	0.1569	0.2020	0.1558
Harris	0.2434	0.2884	0.6867	0.8481

Graphical analysis for the above table has been provided in Chart -2:

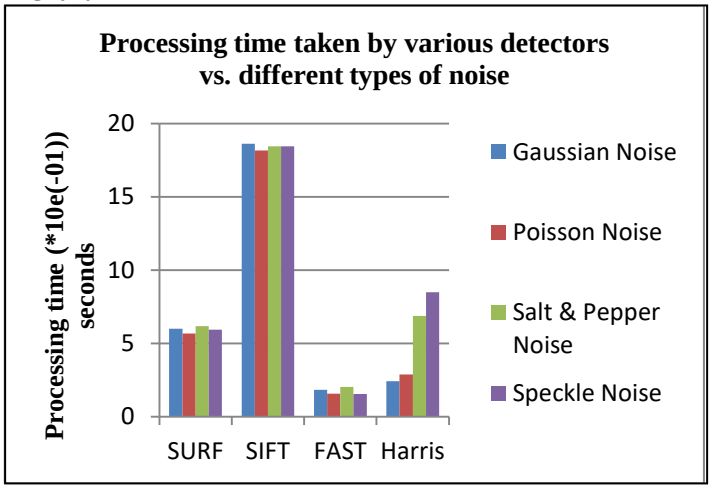


Chart -2: Graphical analysis of processing time taken by each algorithm for feature point detection in case of noisy images.

b) Feature detection and processing time taken in case of filtered images:

Table -3:. Number Of Keypoints Detected By Each Algorithm After Applying Low Pass Adaptive Filter Over Noisy Images

Algorithm	# of keypoints in filtered image	# of keypoints in filtered	# of keypoints in filtered	# of keypoints in filtered

	which consisted of Gaussian noise	image which consisted of Poisson noise	image which consisted of Salt & Pepper noise	image which consisted of Speckle noise
SIFT	1080	1013	1269	1045
SURF	575	587	742	584
FAST	1031	941	3457	917
Harris	601	552	1154	535

Table -4: Number Of Keypoints Detected By Each Algorithm After Applying Low Pass Average Filter Over Noisy Images

Algorithm	# of keypoints in filtered image which consisted of Gaussian noise	# of keypoints in filtered image which consisted of Poisson noise	# of keypoints in filtered image which consisted of Salt & Pepper noise	# of keypoints in filtered image which consisted of Speckle noise
SIFT	3652	2446	6083	3054
SURF	558	554	651	577
FAST	414	366	556	383
Harris	821	610	1132	697

Table -5: Number Of Keypoints Detected By Each Algorithm After Applying Low Pass Gaussian Filter Over Noisy Images

Algorithm	# of keypoints in filtered image which consisted of Gaussian noise	# of keypoints in filtered image which consisted of Poisson noise	# of keypoints in filtered image which consisted of Salt & Pepper noise	# of keypoints in filtered image which consisted of Speckle noise
SIFT	2322	2856	4013	2554
SURF	810	561	679	763
FAST	1321	466	1001	1045
Harris	1592	509	1678	1332

Table -6: Number of Key points Detected By Each Algorithm After Applying Low Pass Median Filter Over Noisy Images

Algorithm	# of keypoints in filtered image which consisted of Gaussian noise	# of keypoints in filtered image which consisted of Poisson noise	# of keypoints in filtered image which consisted of Salt & Pepper noise	# of keypoints in filtered image which consisted of Speckle noise
SIFT	3632	2147	1878	4124
SURF	642	585	586	680
FAST	629	601	672	641
Harris	597	531	599	524

The graphical representation for the feature detected in case of filtered images has been shown below in Chart -2– Chart-6.

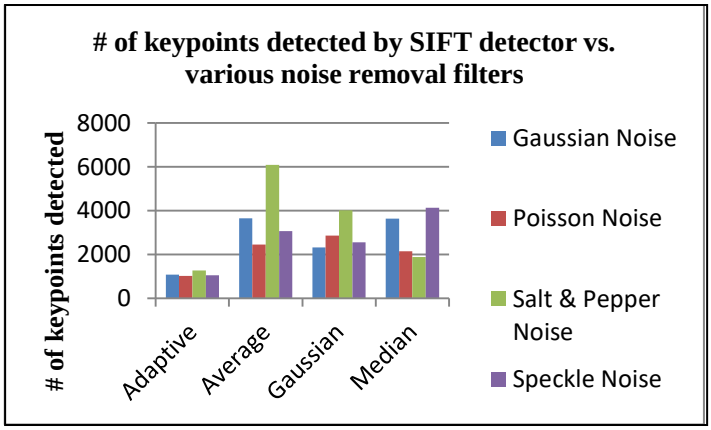


Chart -3: Graphical analysis of keypoint detection by SIFT algorithm in case of filtered images.

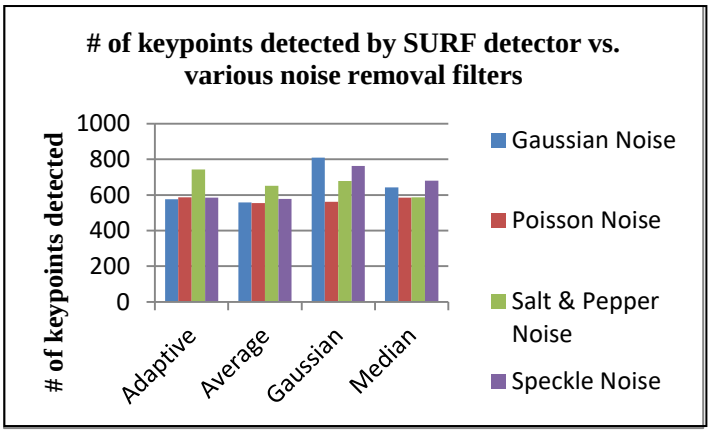


Chart -4: Graphical analysis of keypoint detection by SURF algorithm in case of filtered images.

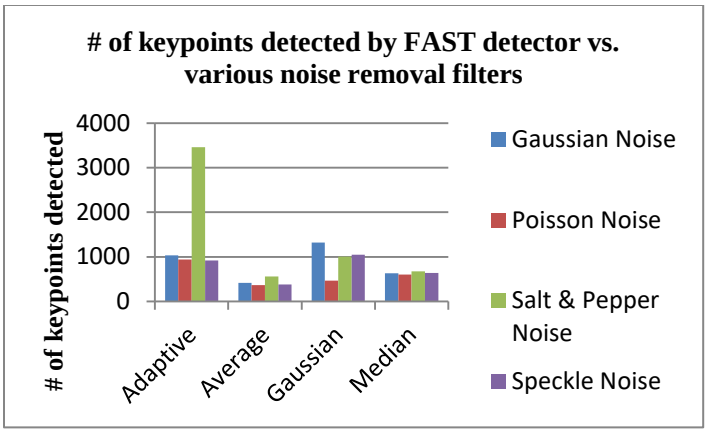


Chart -5: Graphical analysis of keypoint detection by FAST algorithm in case of filtered images.

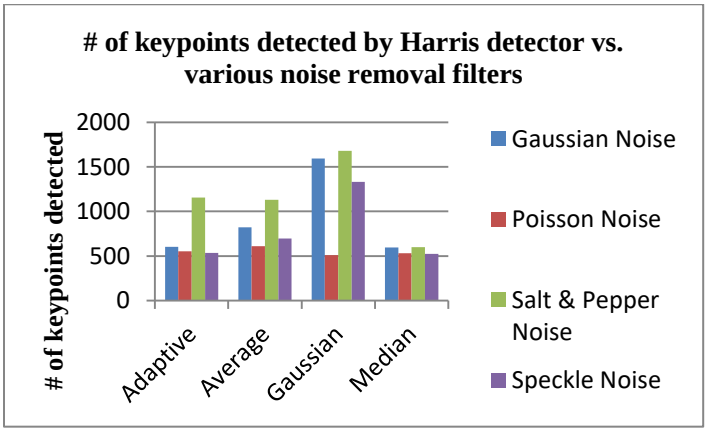


Chart -6: Graphical analysis of keypoint detection by Harris algorithm in case of filtered images.

It can be observed that many features get removed when an image is filtered with a low pass filter. Though few features may be lost, still filtering a captured image is highly recommended and is an important part of image preprocessing. It is also noticed that all the feature detectors detect maximum number of keypoints in an average when an image is filtered with the low pass Gaussian filter.

The processing time for detecting features from filtered images have also been calculated which has been shown below in tabular fashion as well as in a graphical manner.

Table -7: Processing Time Taken By Each Algorithm After Applying Low Pass Adaptive Filter Over Noisy Images

Algorithm	Image with Gaussian noise (in seconds)	Image with Poisson noise (in seconds)	Image with Salt & Pepper noise (in seconds)	Image with Speckle noise (in seconds)
SIFT	1.782	1.682	1.782	1.701
SURF	0.4335	0.4448	0.5405	0.4896
FAST	0.1937	0.1459	0.1990	0.1522
Harris	0.2499	0.2266	0.2043	0.2323

Table -8: Processing Time Taken By Each Algorithm After Applying Low Pass Average Filter Over Noisy Images

Algorithm	Image with Gaussian noise (in seconds)	Image with Poisson noise (in seconds)	Image with Salt & Pepper noise (in seconds)	Image with Speckle noise (in seconds)
SIFT	2.466	2.082	3.076	2.444
SURF	0.4367	0.4368	0.4980	0.4447
FAST	0.1599	0.1566	0.1953	0.1510
Harris	0.2595	0.1907	0.2421	0.2672

Table -9: Processing Time Taken By Each Algorithm After Applying Low Pass Gaussian Filter Over Noisy Images

Algorithm	Image	Image	Image	Image

	with Gaussian noise (in seconds)	with Poisson noise (in seconds)	with Salt & Pepper noise (in seconds)	with Speckle noise (in seconds)
SIFT	2.043	2.163	2.476	2.095
SURF	0.5849	0.4503	0.5103	0.5466
FAST	0.1621	0.1837	0.1480	0.1878
Harris	0.2252	0.2369	0.2273	0.2682

Table -10: Processing Time Taken By Each Algorithm After Applying Low Pass Median Filter Over Noisy Images

Algorithm	Image with Gaussian noise (in seconds)	Image with Poisson noise (in seconds)	Image with Salt & Pepper noise (in seconds)	Image with Speckle noise (in seconds)
SIFT	2.447	2.000	1.925	2.628
SURF	0.4785	0.4425	0.5095	0.5420
FAST	0.1835	0.1430	0.1610	0.1489
Harris	0.2470	0.2093	0.2085	0.1920

The graphical analysis based on processing time taken by each algorithm has been shown below in Chart-7 –Chart-10 .

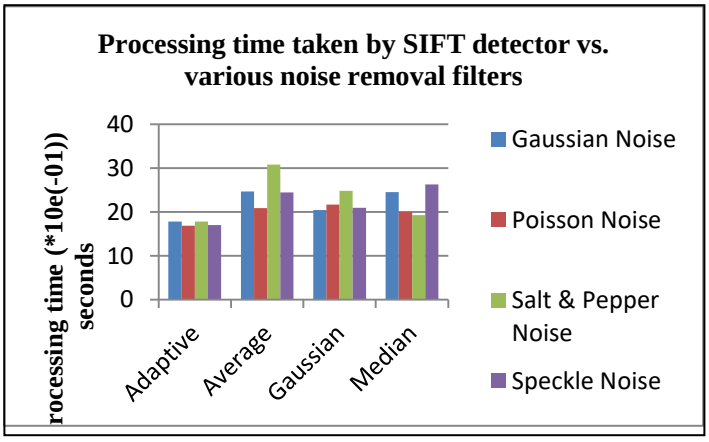


Chart -7: Graphical analysis of processing time taken by SIFT algorithm for feature point detection in case of filtered images

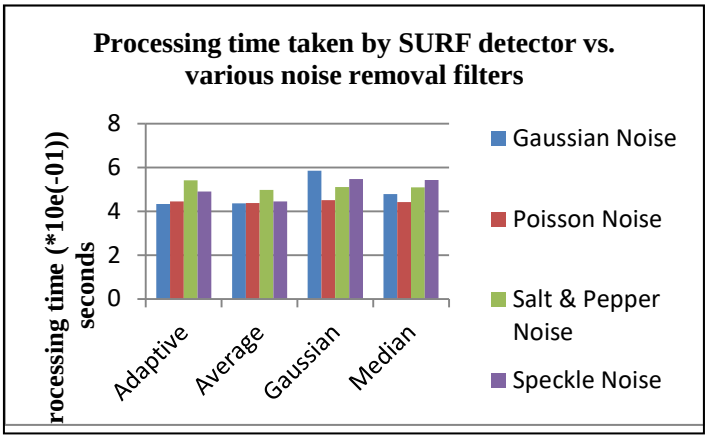


Chart -8: Graphical analysis of processing time taken by SURF algorithm for feature point detection in case of filtered images

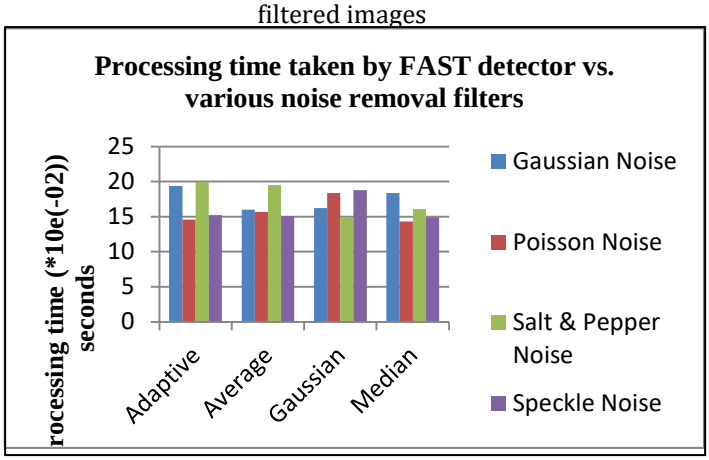


Chart -9: Graphical analysis of processing time taken by FAST algorithm for feature point detection in case of filtered images

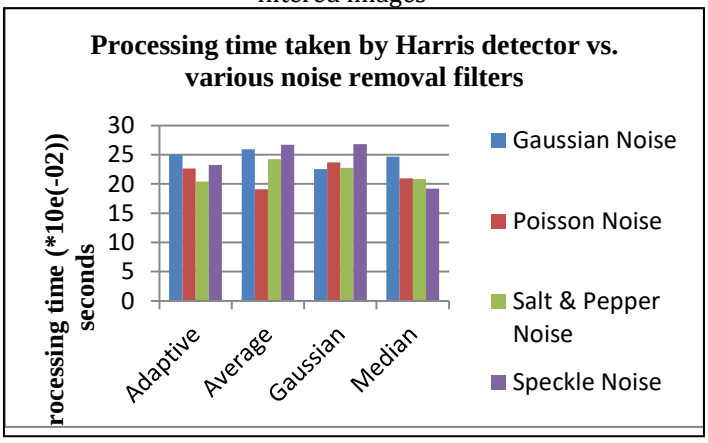


Chart -10: Graphical analysis of processing time taken by Harris algorithm for feature point detection in case of filtered images

Among the four algorithms, SIFT takes the maximum processing time while FAST takes the minimum. If compared, SURF is a better scale invariant feature detection algorithm, while FAST is a better corner detection algorithm.

Not only low pass filtering, but also high pass Gaussian filter has been used for further evaluation of the various filters. The result of high pass filtering has been presented below in tabular as well as graphical manner (in Chart -11, Chart -12).

Table -11: Number Of Keypoints Detected By Each Algorithm After Applying High Pass Gaussian Filter Over Noisy Images

Algorithm	# of keypoints in filtered image which consisted of Gaussian noise	# of keypoints in filtered image which consisted of Poisson noise	# of keypoints in filtered image which consisted of Salt & Pepper noise	# of keypoints in filtered image which consisted of Speckle noise
SIFT	1473	1371	1516	1435
SURF	1574	1301	1460	1498
FAST	11324	4505	9373	6976
Harris	5349	1781	4017	5153

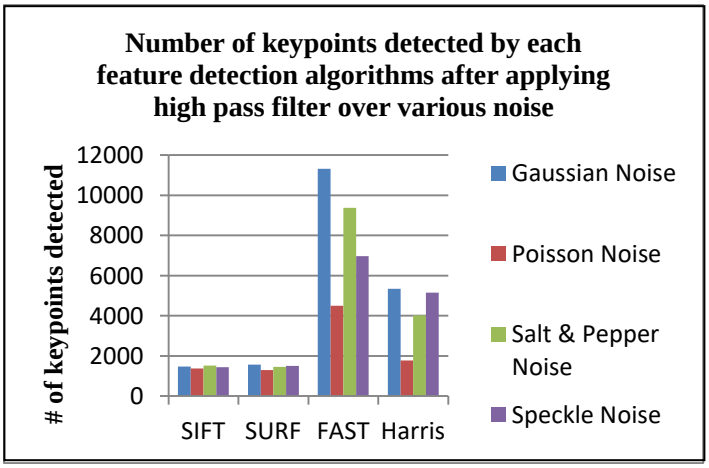


Chart -11: Graphical analysis of keypoint detection by each algorithm in case of high pass filtered images.

Table -12: Processing Time Taken By Each Algorithm After Applying High Pass Gaussian Filter Over Noisy Images

Algorithm	Image with Gaussian noise (in seconds)	Image with Poisson noise (in seconds)	Image with Salt & Pepper noise (in seconds)	Image with Speckle noise (in seconds)
SIFT	2.3012	1.8989	1.9122	1.8421
SURF	1.0184	0.8274	0.9434	0.9473
FAST	0.2243	0.1736	0.2393	0.2052
Harris	0.4138	0.2462	0.2845	0.2931

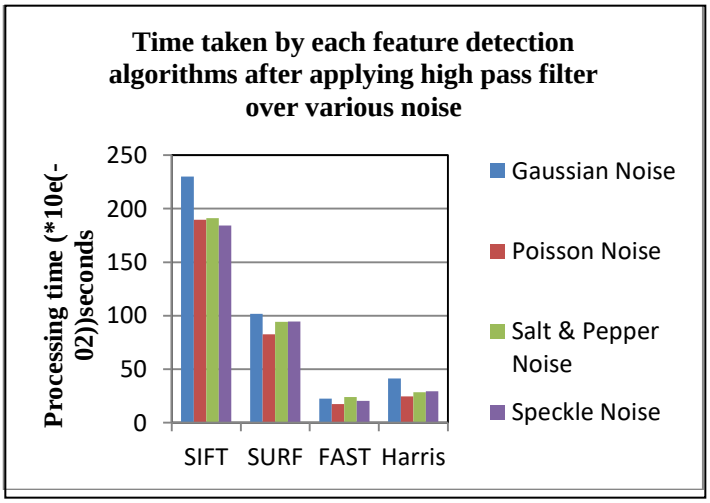


Chart -12: Graphical analysis of processing time taken by each algorithm for feature point detection in case of high pass filtered images

A graphical analysis of the high pass and low pass Gaussian filter can be represented as in Chart -13:

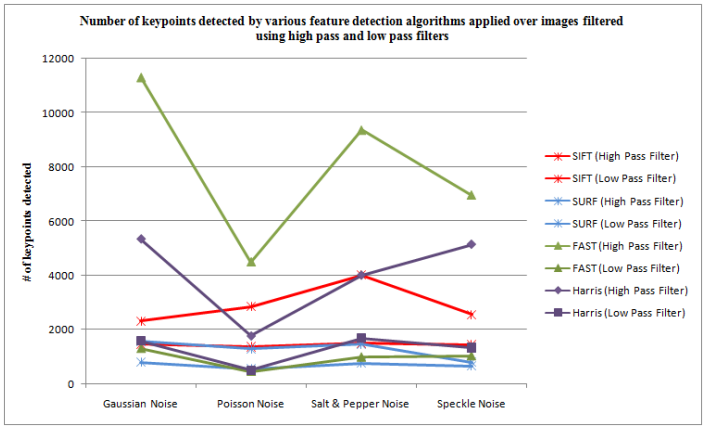


Chart -13: Graphical analysis of processing time taken by each algorithm for feature point detection in case of high pass as well as low pass filtered images

From the above evaluation it can be seen that FAST detects the maximum number of keypoints and also takes the minimum time to process. Thus, FAST is the most efficient algorithm in case of a high pass filtered image.

c) Feature detection, processing time computation and matching performance in case of modified images:

After the preprocessing part, the performance of the feature detection algorithms is analyzed in case of deformed images. The tabular and graphical (Chart-14) representation for the same has been shown below.

Table -13: Number of Key points Detected by Each Algorithm for Modified Images

Algorithm	# of keypoints in input image	# of keypoints in cropped image	# of keypoints in rotated image	# of keypoints in blurred image
SIFT	1391	398	1381	1127
SURF	774	187	782	436
FAST	1379	254	1402	352
Harris	754	225	744	523

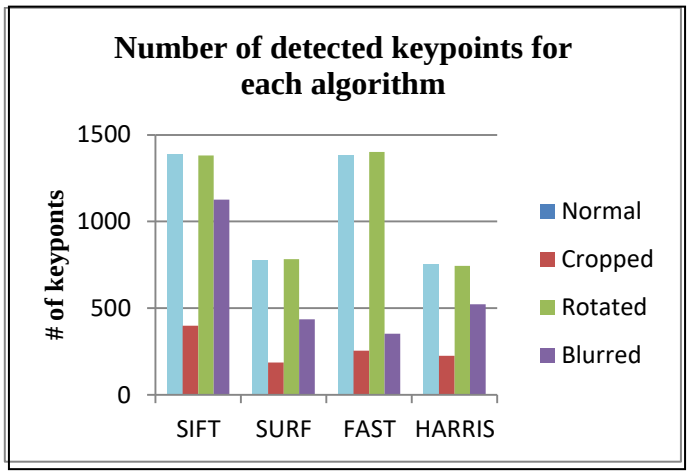


Chart -14: Graphical analysis of keypoint detection by each algorithm in case of modified images.

From the above data and charts, it can be concluded that SIFT is efficient in detecting maximum number of feature points, closely followed by FAST. Also, in case of the transformations, all the algorithms detect maximum keypoints in case of the rotated image. Minimum keypoints are detected in case of cropped image since the area gets reduced; hence the feature points also get reduced.

After detecting and extracting the keypoints, the input image is compared with the modified images to find the matching feature points. Percentage of match is also calculated and plotted for the comparative study as well as shown in Chart-15 and Chart-16.

Table -14: Number of Matched Keypoints Detected By Each Algorithm for Modified Images

Algorithm	Modifications of input image	# of matched keypoints	Matching percentage (%)	Processing time (seconds)
SIFT	Cropped	339	24.37	1.6033
	Rotated	1148	82.53	2.903
	Blurred	235	16.89	2.156
SURF	Cropped	184	23.77	0.2814
	Rotated	664	85.78	0.3352
	Blurred	173	22.35	0.2942
FAST	Cropped	228	16.53	0.446

	Rotated	1067	77.37	0.765
	Blurred	19	1.37	0.430
Harris	Cropped	131	17.37	0.4342
	Rotated	494	65.51	0.4712
	Blurred	12	1.59	0.4589

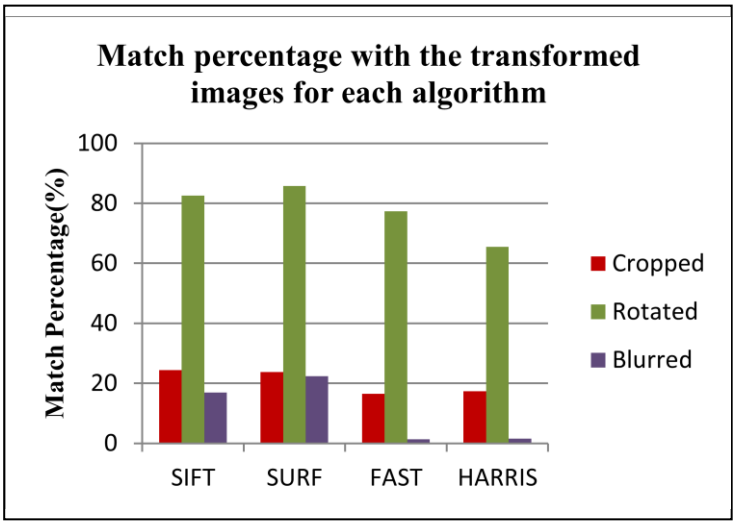


Chart -15: Graphical analysis of matching percentage of each algorithm in case of modified images.

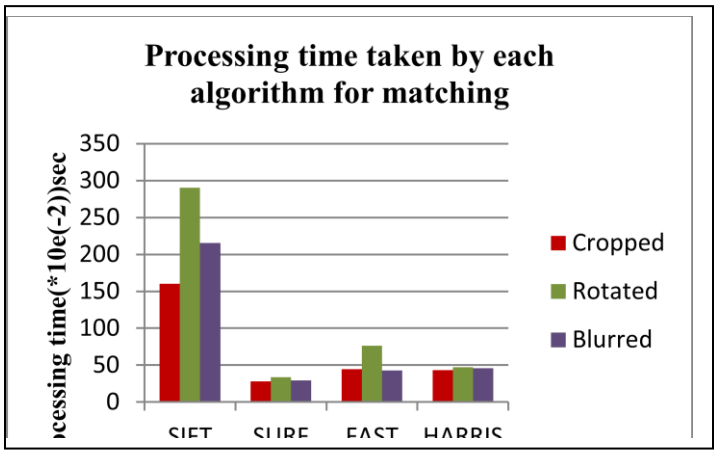


Chart -16: Graphical analysis of processing time taken by each algorithm in case of modified images.

From the above data and chart, it can be seen that SIFT and SURF almost possess the same amount of efficiency while FAST and Harris is slightly less capable. But according to the evaluation based on processing time, SURF, FAST and Harris are way faster that SIFT algorithm. Since SURF takes the minimum processing time and is capable of finding second highest number of matching points, hence it can be concluded that SURF is the most efficient.

d) Feature detection, processing time computation and matching performance in case of a set of images:

Apart from finding matches with transformed images, another study is also performed on how much potent these algorithms are in detecting matched keypoints from within a set of images along with the input image.

Table -15: Number of Keypoints Detected By Each Algorithm For Varied Number Of Image Sets.

Algorithm	# of keypoints in input image	# of keypoints in set of 2 images	# of keypoints in set of 4 images	# of keypoints in set of 6 images
SIFT	1391	826	1489	2069
SURF	774	297	595	847
FAST	1379	411	725	1091
Harris	754	157	365	539

The number of keypoints detected is maximum in case of SIFT, followed by FAST, SURF and lastly Harris. Also, the number of detected keypoints increases as the number of images in the data set increases.
The matched feature points of the input image within the various sets of images along with the processing time have been presented below in tabular and graphical (Chart-17 and Chart-18) form.

Table -16: Number Of Matched Keypoints Detected By Each Algorithm For Varied Image Sets.

Algorithm	# of images in dataset	# of matched keypoints	Matching percentage (%)	Processing time (seconds)
SIFT	2	228	16.39	1.697
	4	212	15.24	2.151
	6	219	15.74	7.416
SURF	2	74	9.56	0.298

	4	69	8.91	0.338
	6	69	8.91	0.315
FAST	2	0	0	0.4242
	4	0	0	0.5904
	6	0	0	0.3779
Harris	2	0	0	0.644
	4	0	0	0.373
	6	0	0	0.406

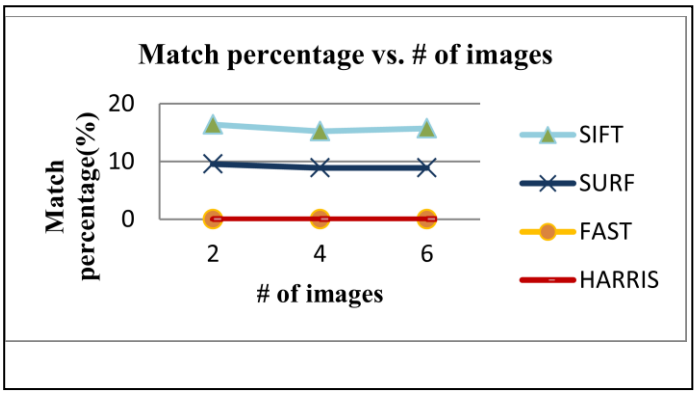


Chart -17: Graphical analysis of matching percentage taken by each algorithm in case of various image sets.

The matching percentage is higher in case of SIFT, closely followed by SURF. Since SIFT detected the maximum number of keypoints, hence its capable of matching maximum keypoints within the group of images. FAST and Harris however could not detect any matches from the set of images.

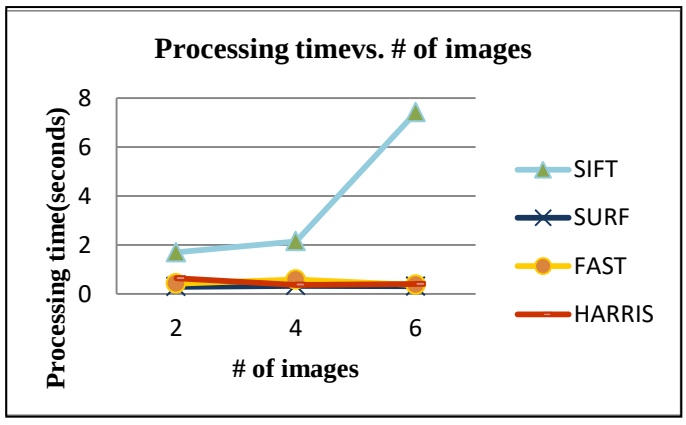


Chart -18: Graphical analysis of processing time taken by each algorithm in case of various image sets.

After plotting the performance of the algorithms on the basis of processing time, it can be observed that SIFT took the maximum processing time, which also increases as the number of images within the data set increases. SURF is the fastest and remains almost same with the increase in the number of images within the image set.

From the exhaustive analysis of the preprocessing, feature detection, feature extraction and matching, the results can be summarized as:

- ❖ SIFT is capable of detecting the maximum number of keypoints in most cases, closely followed by FAST.
- ❖ SURF takes lower processing time than SIFT, both of which are scale invariant feature detection algorithms.
- ❖ FAST takes lower processing time than Harris, both of which are corner detection algorithms.
- ❖ Gaussian filter preserves maximum feature points in average, while filtering an image.
- ❖ Low pass filter results in smoothening of the image while high pass filter results in sharpening of the image.

6. PRELIMINARY TASKS FOR 3D RECONSTRUCTION

After the elaborate and exhaustive study about feature detection and carrying out the experiment of selecting the best feature detection algorithm, the preliminary approach of 3D reconstruction from a video taken by a single uncalibrated camera have been explained.

SURF algorithm is preferred to carry out the experiment, since it is fairly efficient.

From the related works mentioned above on 3D reconstruction, the steps involved in the process can be listed out as:

- ✓ Take a video with a single uncalibrated camera of a scene.
- ✓ Segregate the video file into number of image frames.
- ✓ Detect and extract features from each frame.
- ✓ Find matching correspondences within the adjacent image frames.
- ✓ Stitch the image frames.
- ✓ Find the disparity between the frames and display the disparity map.
- ✓ Calculate the depth from the 2D image frames.
- ✓ 3D reconstruction by point cloud technique.
- ✓ Finally, texture mapping to get the 3D scene.

In this paper, the task till image stitching and panorama generation has been accomplished. An input video file [27] (snapshot shown in Fig. 42.), taken by a single moving camera has been considered. The video is then divided into its corresponding frames (Fig. 43.). The frame rate is 25 fps, thus the video of 12 seconds got partitioned into 307 frames.

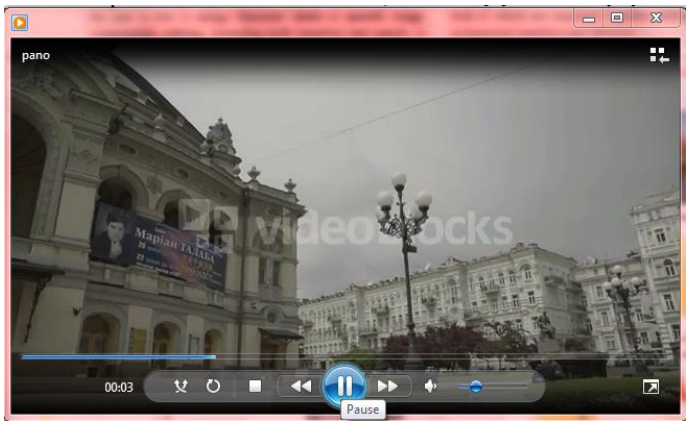


Fig- 42: Snapshot of the input video clip.

Fig- 43: Few overlapping frames segregated from the video file.

After getting the image frames, the preprocessing part, i.e. filtering the images is carried out over each frame. Low pass Gaussian filter has been used in this case represented in Fig. 44.

Fig-44: Filtered frames of the video file.

Next, the SURF feature detection algorithm is run to detect and extract features from the images. The detected features in the frames can be shown in the figures below in Fig. 45.

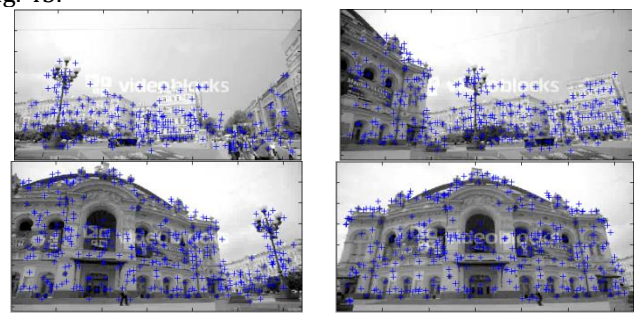


Fig-45: Features detected using SURF descriptor.

Next step is finding out the matching correspondences within the adjacent frames (in Fig. 46). SURF algorithm has been employed to find the matches, after which the images are registered and the frames are then stitched to one another, forming a panorama.

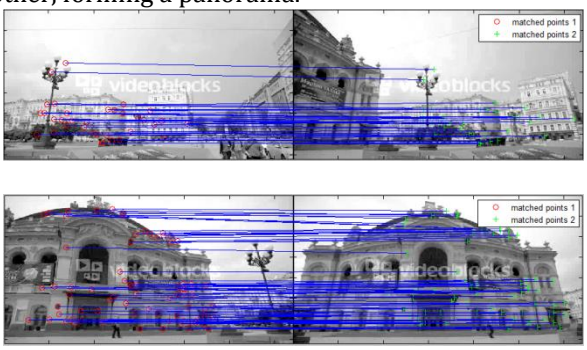


Fig-46: Matched features using SURF descriptor.

The stitched panorama has been represented in the following Fig. 47.

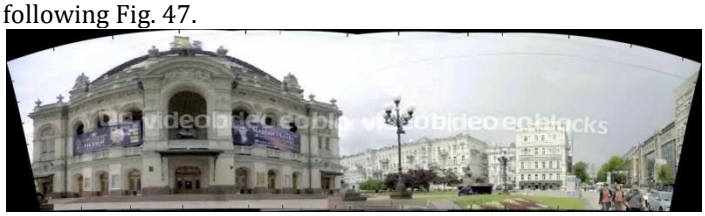


Fig-47: The panoramic image.

7. CONCLUSIONS AND FUTURE WORKS

The elaborated comparative study is done with the purpose of evaluating the performance and efficiency of the existing feature descriptors so that an efficient algorithm can be chosen to carry out the 3D reconstruction process. Preprocessing has been carried out using various filters to

remove noises from an image. From the results a suitable filtration method can be adopted according to the need. We have selected low pass Gaussian filter for carrying out the preprocessing part. Comparisons have also been done based on various transformations of the input image as well as on a varied set of images. Evaluation on the basis of match percentage and processing time has been shown. From the study it can be concluded that SURF is efficient as well as faster than rest of the algorithms specified. SIFT and FAST are efficient in finding feature points as well as matching, but takes comparatively higher processing time. Harris, though operate faster than SIFT but is less efficient in detecting and extracting keypoints. The study thus provides an individual as well as comparative assessment. SURF feature detector is applied for detecting, extracting and matching the feature points within the image frames segregated from the video scene. Finally, a panorama from the stitched images gets formed.

The next move includes finding the disparity among the frames and plots the disparity map. Depth calculation is next major and challenging step since the camera is not calibrated. Also, no such infrared laser has been used, which is a method by which Kinect calculate the depth. Next step is to create a wire structure using point cloud. Finally, texture mapping is to be done to get the final 3D view of the scene.

REFERENCES

[1] B. Chitradevi and P.Srimathi, "An Overview on Image Processing Techniques", International Journal of Innovative Research in Computer and Communication Engineering, ISSN ONLINE(2320-9801) PRINT (2320-9798).

[2] Ajay Kumar Boyat and Brijendra Kumar Joshi, "A Review Paper: Noise Models In Digital Image Processing", Signal & Image Processing : An International Journal (SIPIJ) Vol.6, No.2, April 2015.

[3] Aziz Makandar and Bhagirathi Halalli, "Image Enhancement Techniques using Highpass and Lowpass Filters", International Journal of Computer Applications (0975 – 8887) Volume 109 – No. 14, January 2015.

[4] Pawan Patidar and Sumit Srivastava, "Image De-noising by Various Filters for Different Noise", International Journal of Computer Applications (0975 – 8887) Volume 9– No.4, November 2010.

[5] Wikipedia: "Adaptive Filter".

[6] Sunil Kopparapu and M Satish, "Optimal Gaussian Filter for Effective Noise Filtering", 12 Jun 2014.

[7] Somajyoti Majumder, "Sensor Fusion and Feature Based Navigation for Subsea Robots", The University of Sydney, August 2001.

[8] Wikipedia: "Scale-invariant feature transform".

[9] David G. Lowe, "Distinctive Image Features from Scale-Invariant Keypoints", International Journal of Computer Vision, January 5, 2004.

[10] Jacob Toft Pedersen, "Study group SURF: Feature detection & description", SURF: FEATURE DETECTION & DESCRIPTION, Q4 2011.

[11] Deepak Geetha Viswanathan, "Features from Accelerated Segment Test (FAST)".

[12] E. Rosten and T. Drummond, "Machine learning for high speed corner detection," in 9th Euproean Conference on Computer Vision, vol. 1, 2006, pp. 430–443.

[13] Wikipedia: "Corner Detection (The Harris & Stephens / Plessey / Shi–Tomasi corner detection algorithms)".

[14] Konstantinos G. Derpanis , "The Harris Corner Detector", October 27, 2004.

[15] Jan-Michael Frahm , Marc Pollefeys , Brian Clipp , David Gallup , Rahul Raguram , ChangChang Wu , and Christopher Zach, "3D Reconstruction Of Architectural Scenes From Uncalibrated Video Sequences".

[16] Bing Han, Christopher Paulson, and Dapeng Wu, "3D Dense Reconstruction from 2D Video Sequence via 3D Geometric Segmentation".

[17] Tomas Rodriguez, Peter Sturm, Marta Wilczkowiak, Adrien Bartoli, Mathieu Peronnaz, et al.. VISIRE. Photorealistic 3D Reconstruction from Video Sequences. International Conference on Image Processing (ICIP 2003), Sep 2003, Barcelona, Spain. IEEE Signal Processing Society, 2, pp.III - 705-8, 2003, .

[18] Dang Trung Kien, "A Review of 3D Reconstruction from Video Sequences", Intelligent Sensory Information Systems, technical report series, 2005.

[19] Jason Repko, Marc Pollefeys, "3D Models from Extended Uncalibrated Video Sequences: Addressing Key-frame Selection and Projective Drift".

[20] Evren Đmre, Sebastian Knorr , A. Aydın Alatan, and Thomas Sikora, "Prioritized Sequential 3d Reconstruction In Video Sequences With Multiple Motions".

[21] M. Pollefeys, D. Nist´er , J.-M. Frahm, A. Akbarzadeh, P. Mordohai, B. Clipp, C. Engels, D. Gallup, S.-J. Kim, P. Merrell, C. Salmi, S. Sinha, B. Talton, L. Wang, Q. Yang , H. Stew´enius, R. Yang , G. Welch, H. Towles, "Detailed Real-Time Urban 3D Reconstruction From Video".

[22] Luo Juan, Oubong Gwun, "A Comparison of SIFT, PCA-SIFT and SURF", International Journal of Image Processing, August 31, 2009.

[23] Krystian Mikolajczyk, Cordelia Schmid, "A Performance Evaluation of Local Descriptors", IEEE Transactions On Pattern Analysis And Machine Intelligence, VOL. 27, NO. 10, October 2005.

[24] Raman Maini, Dr. Himanshu Aggarwal, "Study and Comparison of Various Image Edge Detection Techniques", International Journal of Image Processing (IJIP), Volume (3) : Issue (1).

[25] Pooja Ghosh, Achala Pandey and Umesh C. Pati, "Comparison of Different Feature Detection Techniques for Image Mosaicing", ACCENTS Transactions on Image Processing and Computer Vision, Volume-1 Issue-1 November-2015.

[26] Khairulmuzzammil Saipullah, Nurul Atiqah Ismail, Ammar Anuar, Nuraishah Sarimin, "Comparison Of Feature Extractors For Realtime Object Detection On Android Smartphone" , Journal of Theoretical and Applied Information Technology, January 10, 2013.

[27] https://www.videoblocks.com/video/views-of-kiev-panorama-of-the-building-of-the-national-opera-of-ukraine-city-center-bpzt_evc8iqf3xf5e/

BIOGRAPHIES

Mouli Laha has completed her B.Tech in Computer Science and Engineering from West Bengal University of Technology (2014). She worked as a Research Fellow at CSIR-CMERI under the Ministry of Science, Government of India. Next, she pursued her M.Tech in Computer Science and Engineering from Indian Institute of Technology (Indian School of Mines), Dhanbad (2018). She has qualified GATE in 2014 and 2018, UGC-NET in 2018. Currently, she is serving at National Informatics Centre under Ministry of Electronics and Information Technology, Government of India.