

Image quality evaluation: evaluation of the image quality of actual images by using machine learning models

Shiva Shankar Reddy¹, Veeranki V. R. Maheswara Rao², Kalidindi Sravani¹, Silpa Nrusimhadri²

¹Department of Computer Science and Engineering, Sagi RamaKrishnam Raju Engineering College (A), Bhimavaram, India

²Department of Computer Science and Engineering, Shri Vishnu Engineering College for Women (A), Bhimavaram, India

Article Info

Article history:

Received Feb 26, 2023

Revised Aug 30, 2023

Accepted Sep 28, 2023

Keywords:

Deep learning

Image feature

Image quality evaluation

Inlier extraction

Machine learning

ABSTRACT

Evaluating image features is a significant step in image processing in applications like number plate detection, vehicle tracking and many image processing-based applications. Image processing-based applications need accurate parts to get the best outcomes. Feature detection is done based on various feature detection techniques. The proposed system aims to get the best feature detector based on the input images by evaluating the image features. For assessing the image features, the proposed system worked on various descriptors like oriented FAST and rotated brief (ORB), learned arrangements of three patch codes (LATCH), binary robust independent elementary features (BRIEF), and binary robust invariant scalable keypoints (BRISK) to extract and evaluate the features using K-nearest neighbor (KNN)-matching and retrieve the inliers of the matching. Each descriptor produces different matching features and inliers; with the matchings and inliers, the inlier ratio calculates to show the analysis. To increase performance, we also examine adding depth information to descriptors.

This is an open access article under the [CC BY-SA](#) license.



Corresponding Author:

Shiva Shankar Reddy

Department of Computer Science and Engineering, Sagi RamaKrishnam Raju Engineering College (A)
Bhimavaram, Andhra Pradesh, India

Email: shiva.csesrkr@gmail.com

1. INTRODUCTION

Image quality evaluation (IQE) enables computers to perceive picture quality in a way that matches human perception. Visual quality may be quantified and improved for numerous vision activities. The accessibility to the new reference picture determines the IQE approach: full-reference, reduced-reference, or no-reference. Despite the extensive research on the correlation between picture attributes and perceived quality, no study has yet examined the potential quality of a restored image. Evaluating the image's features is a crucial step in image processing. Because these applications must handle the best feature point descriptors, there is an increasing demand for best descriptors that are quick to calculate and match, memory-efficient, and accurate [1]. They used various tasks such as comprehensive baseline stereo matching, panorama stitching, 3D scene reconstruction, object, scene, texture, and gesture recognition [2]. Incorporating key features specific to their intended use is critical to the success of mobile applications [3]. In image processing, different methods, and techniques extract the features. Every application of image processing is gone through removing the elements of the image to get accurate results [4]. The matching method uses point-to-point alignments to speed up branch-and-bound [5]. A wide variety of applications make use of these pattern-matching algorithms. Feature identification and extraction methods increased as image analysis got more prominent. Scale-invariant feature transform (SIFT) [6], speeded up robust features

(SURF), binary robust invariant scalable keypoints (BRISK), and oriented FAST and rotated brief (ORB)[7] descriptors detect vital points and feature matching, as shown in Figure 1.

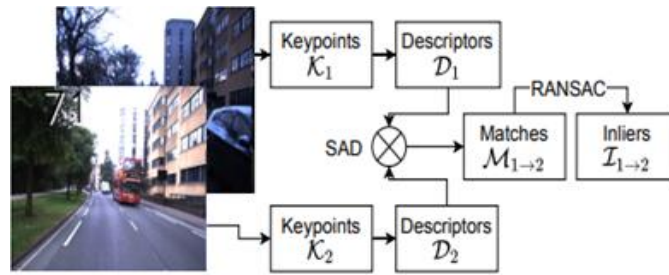


Figure 1. Inlier extraction of the real-world images

The homography matrix is a primary step of this evaluation to find the pure matches between images called inliers. The inliers calculation is the primary aspect for evaluating the image features. Homography matrix and image matching calculate inliers [8]. Matching involves in the system with the image keypoints and descriptors. These keypoints and descriptors come under feature extraction with the help of detectors. For the best image key points and descriptors, the ORB and boosted efficient bi-nary local image descriptor (BEBLID) detectors performed and various types of detectors like learned arrangements of three patch codes (LATCH), fast retina keypoint (FREAK), BRISK, and BOOST [9] on the different real-world images, which include the orientation and variations of the pictures. Through those images, we extract the keypoints and descriptors and find the matching using the K-nearest neighbor (KNN)-matching for the pictures; with those matches, we calculate the inliers [10].

Calonder *et al.* [1] proposed binary robust independent elementary features (BRIEF), a very fast local binary descriptor, by testing with the descriptors like shift, surf, u-surf and u-shift and found the BRIEF as a fast binary descriptor. Kaneva *et al.* [2] suggested photorealistic virtual environment picture feature evaluation. Designing and comparing feature descriptions requires assessing their performance relative to those modifications. They studied descriptions' illumination, scene, and viewing sensitivity. Controls truth data from many locations considered under different cameras and lighting conditions are needed. Gathering such knowledge in real life is complicated. Igler [3] introduced a lightweight process model to help identify virtual user interface and application logic properties. The iterative, incremental process approach is based on engineering principles for software product lines and combined with design science research. Each cycle involves the development of several prototype versions that the client evaluates. Prototypes employ feature models.

Remmen *et al.* [4] examined how real-world perturbations affect six traditional feature-based approaches and two convolutional neural network (CNN) methods over a year, a month, and a year. The number of recovered inliers and processing times utilized to assess strategies for low, medium, and high-complexity instances. Babu *et al.* [5] recognized the face expression using bezier curves and 2D image processing. Rosten *et al.* [6] suggested extra processing; efficiency determines whether the detector can function at the frame rate. This work presents a novel feature detection heuristic and a machine learning-based feature detector that can thoroughly analyze live phase alternating line (PAL) video in less than 5% of the processing time. Then generalize the detectors to tweak them for repeatability and efficiency. Lepetit and Fua [7] suggested a statistical learning-based solution for 3D object identification and posture estimation that achieves speed and reliability at runtime. The wide-baseline matching problem, which this covers, has been recast as a classification issue that can be solved using fast and simple randomized trees. Winder and Brown [8] use an interest point descriptor for 3D reconstruction and age matching. They examined descriptor algorithm components and evaluated their combinations. SIFT, gradient location-orientation histogram (GLOH), and spin images are published descriptors that fit our system. To learn appropriate parameter choices for each candidate method, train on patches from a multi-image 3D reconstruction with reliable ground-truth matching. Jakubovic and Velagic [9] suggested feature matching and object recognition in two photos are solved using brute-force matches in this study. The recommended feature identification and descriptor extraction framework utilized ORB, BRISK, SIFT, and SURF. Features matched using brute-force and the KNNs method. The robust random sample consensus (RANSAC) technique employs acquired matches to estimate the transformation between two successive photos. Porav *et al.* [10] suggested that computer vision tasks become substantially more challenging when rain and lenses are present because local picture distortions start to appear. Create a pre-processing filter to decrease raindrops on lenses in this article. Sun *et al.* [11] converted daytime images to nighttime ones for adequate semantic segmentation. Performance depends on the percentage of synthetic

nighttime photos, as shown in this research. The optimal point is consistently high performance throughout the day and night.

Mikolajczyk and Schmid [12] suggested the Harris-Affine detector retrieved local interest region descriptors and compared them to shift and distributed descriptors. Jing *et al.* [13] told colored binary robust invariant scalable key points (CBRISK). They're coloured binary key points in this publication. Due to the relevance of colour information in vision applications, he created CBRISK, a revolutionary key point identification and description approach that considers colour. Given its significance in vision applications, this paper introduces CBRISK, a novel approach for incorporating colour information into crucial point detection and description. The recommended solution uses photometric invariant colour space key points instead of greyscale intensity pictures [14]. The whole feature point handling pipeline, including detection, orientation estimation, and feature description, is implemented using a unique deep network architecture he describes. Li *et al.* [15] suggested asymmetric generative adversarial networks (GAN) for unpaired image-to-image translation. They claimed that asymmetric GAN offers a superior approach for odd image-to-image translation in asymmetric domains.

Hua *et al.* [16] introduced discriminant embedding for local image descriptors. SIFT and GLOH are very strong for image matching and visual recognition. However, SIFT descriptors parameterize in 128-dimensional spaces. Winder *et al.* [17], often high-dimensional, investigated dimension reduction. Strecha *et al.* [18] studied using a simple method to produce a binary string using SIFT descriptor. With a tiny description, this technique matches well. For large-scale keypoint retrieval applications, our binary descriptor beats SIFT and digital accessible information system (DAISY) in the low false positive range.

Ozuysal *et al.* [19] show that a simple, efficient, and resilient solution solves by phrasing the issue in a naïve Bayesian classification framework. This classifier combines hundreds of essential binary variables and class posterior probability to differentiate patches around critical points. Assuming independence across feature sets makes the task computationally tractable.

According to Rosten and Drummond [20], SIFT, Harris, and smallest univalue segment assimilating nucleus (SUSAN) provide high-quality features, but they are too computationally expensive for real-time applications. Machine learning uses to develop a feature detector that can thoroughly analyze live PAL video in less than 7% of the time available. "SURF" was suggested by Bay *et al.* [21]. They provided a fast and performance scale and rotation-invariant interest point detector and descriptor. They asserted that the Laplacian-based indexing method speeds up the matching step without sacrificing performance. Agrawal *et al.* [22] suggested picture matching goal is to detect similarities between two photographs of the same setting. Object identification, image indexing, structure from motion, and visual localization are some uses of this core computer vision topic.

Baumberg [23] presented reliable feature matching over widely dispersed images. Many orientations compare by using different techniques. The conclusion feature matching procedure optimizes for structure-from-motion applications that overlook unreliable matches at the price of fewer feature matches. Lowry *et al.* [24] overview of works on visual location identification. This article discusses place recognition in animals, how "place" defines in robotics and the main components of a place recognition system. Long-term robot operations have shown that changing appearance may cause visual place identification failure. Thus, they discussed how to place recognition systems could implicitly or explicitly account for an appearance change. Shankar *et al.* [25] used a technique that reduced the noise in portable grey map (PGM) images by using object-oriented fuzzy filters and analyzed the performance using social group optimization (SGO) and accelerated particle swarm optimization (APSO) [26].

Szeliski [27] have one of the most popular consumer uses of picture registration, and blending is stitching many photos together to create stunning, high-resolution panoramas. They discussed direct intensity-based and feature-based registration methods, global and local alignment, and high-accuracy correspondences between overlapping pictures using panoramic image stitching motion models. After that, they discuss several other compositing approaches, such as multi-band and gradient-domain mixing, as well as how to get rid of blurry and ghosted pictures. These approaches may build high-resolution panoramas for static or interactive viewing. Devareddi *et al.* [28] have done their work on image segmentation on the scanned document using neural networks. Alahi *et al.* [29] suggested "Freak: fast retina keypoint" in 2012. In this study, the learning stage uses to select the most significant. Gaussian difference fits one probable explanation of human visual system resource optimization. This research focuses on selecting key pairings for future high-level applications like object recognition.

2. METHOD

In regards to evaluating the images, the suggested model has utilized certain objectives. We used four objectives in this proposed work. Figure 2 illustrates the workflow's modules. As mentioned in the following sections, the whole process was employed to accomplish the work.

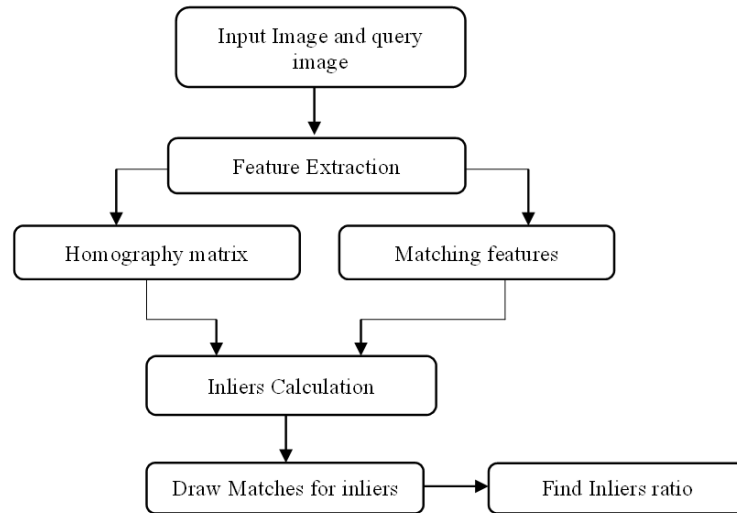


Figure 2. Flow of proposed work

2.1. Objectives

The aim of the research is to evaluate the image quality of actual images. This aim can be reached with the following objectives: i) keypoints and descriptors detection, ii) matching descriptors and finding inliers, iii) draw the matches for inliers, and iv) calculate the inlier ratio.

2.2. Work flow

Here Figure 2 shows the entire work flow of the method. After acquiring the images from the datasets, extract the features from them and classify into homography matrix and matching features. After that calculate inliers. From Inliers match the consider the draw matches and finally find the inliers ratio.

2.2.1. Input image and query image

The system requires an input image and a query image. The appropriate and efficient images as input, while the deviating or similar image was the query image. To read the input and query image, we need the “OpenCV” tool, and we use the image read function to read the pictures, as shown in Figures 3(a) and (b).

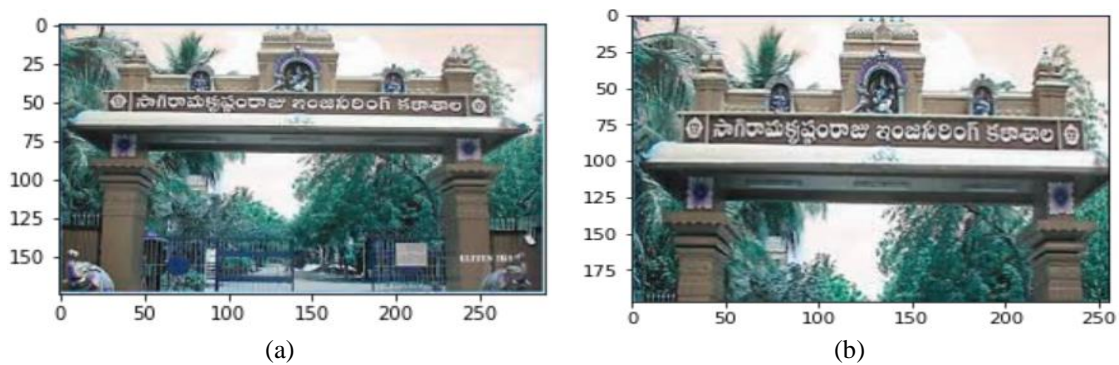


Figure 3. Image reading function as shown in; (a) input image and (b) query images

2.2.2. Feature extraction

Feature extraction involves finding the keypoints and descriptors of the input and query image using feature detectors. For this evaluating feature, we worked on efficient detectors like ORB, BRIEF, BRISK, FREAK, BEBLID, and LATCH. Each detector produces different keypoints and descriptors.

- ORB: it is the result of combining the FAST keypoint detector with the BRIEF descriptor. The ORB takes the query image and converts it to grayscale. The detected keypoints are given as input to the BRIEF to find the descriptor. Compute the descriptors belonging to both pictures.

Finding key points using the FAST detector: the FAST algorithm finds the feature points of the image. Store a vector for each feature point's 16 pixels. Each feature point's 16 pixels uses to calculate the feature vector p . These 16 pixels have three states:

$$S_p \rightarrow x = \begin{cases} d, I_p \rightarrow x < I_p - t \text{ (Darker)} \\ s, I_p - t < I_p \rightarrow x < I_p + t \text{ (Similar)} \\ b, I_p + t < I_p + t \text{ (Brighter)} \end{cases}$$

Depending on these conditions, feature vector P partitioned into three subsets P_d, P_s, P_b . Keypoint k_p and the statement is untrue if p is not an exciting point and accurate if p is? entropy calculates the interesting point.

Finding the descriptor with BRIEF with FAST key points:

- BRIEF: this descriptor uses to detect the descriptors of the input image. In the BRIEF descriptor, the input image is taken and added with scale invariance and rotational invariance, but in this ORB, we use the FAST key points. Those keypoints generated with scale invariance and rotational invariance. After detecting key issues, we find patches to identify descriptors. Patches are squares that surround key points. To build the binary string representing a region around a key point, we must go through all pairs and write 1 in the binary string if the point p_1 has a higher intensity than point p_2 . The descriptors are created by using:

$$Find(p) = \sum_{1 \leq i \leq n_d} 2^{i-1} T(p; x_i, y_j)$$

The ORB produces the descriptors in an n -dimensional array of the input and query image.

- BRISK: BRISK mainly builds the descriptors with long and short pairs. Long pairs use in BRISK to determine the orientation, and short pairs use for the intensity comparisons that create the descriptor. Comparisons of intensities must execute to construct the descriptor. The BRISK compute the exposure with the represented formula:

$$g(p_i, p_j) = (p_i, p_j) \frac{I(p_j, \sigma_j) - I(p_i, \sigma_i)}{\|p_j - p_i\|^2}$$

Where 'i' is the Gaussian-smoothed intensity of the sample point by the appropriate standard deviation, and $g(p_i, p_j)$ is the local gradient between the sampling pair. It checks if the first point in each short pair has a higher smoothed intensity than the second point. BRISK builds the description using only short pairs. BRISK compares the collection of short pairings by rotating them by the calculated orientation:

$$b = \begin{cases} 1, I(p_j^\alpha, \sigma_j) > I(p_i^\alpha, \sigma_i) \\ 0, otherwise \end{cases}$$

- FREAK: FREAK descriptor is a binary descriptor based on brightness comparison tests at many sample sites around a keypoint. The FREAK descriptor samples field centres. points $p = (p_i, p_j)$, where $i, j \in \{1, 2, \dots, N\}$ and i not equal to j . Binary encoded intensity comparison $s(p_a)$ is the FREAK technique.

$$s(p_a) = \begin{cases} 1, if (P_i > P_j) \\ 0, otherwise \end{cases}$$

To detect the descriptors, FREAK uses the represented mathematical equation:

$$F = \sum_{0 < a < N} 2^a \cdot s(p_a)$$

- BEBLID: it is boosting based binary descriptor. It uses to find the descriptors and keypoints from any detectors using the scale_factor argument and some difference of mean grey values in various regions of the images using detector keypoints. This BEBLID descriptor uses for matching and retrieval problems of the images. Scale_factor: it used to adjust the window around detected key points; i) 1.00f scale for ORB keypoints; ii) 6.75f scale for SIFT detected keypoints; iii) 6.25f default scale for KAZE, SURF detected

keypoints; and iv) 5.00f ranking for BRISK, FAST keypoints. A discriminative and fast-to-compute $f(x)$ is BEBLID's secret to efficiency. $f(x)$ is our function for removing irrelevant features.

$$f(x; p_1, p_2, s) = \frac{1}{s^2} \left[\sum I(q) - \sum I(r) \right]$$

Where $R(p, s)$ is a square box of size s that is centre at pixel p , and $I(t)$ is the grey value at pixel t . f calculates the difference between $R(p_1, s)$ and $R(p_2, s)$'s grey values.

- LATCH: binary descriptors evaluate the similarities and differences of image patch triplets. LATCH analyses three-pixel patch intensity to create a single bit in the binary string encoding the patches. At the expense of a little increase in runtime processing, it outperforms current options by a large margin. LATCH descriptor formula:

$$G(W, S_t) = \begin{cases} 1, & \text{if } \|P_{t,a} - P_{t,1}\|_{2_F} > \|P_{t,a} - P_{t,2}\|_{2_F} \\ 0, & \text{Otherwise} \end{cases}$$

2.2.3. Homography matrix

A perspective change between two planes discovered. It transforms points in one picture into corresponding points in another image. Vector matrix of the original plane's points, srcPoints . RANSAC computes a homography matrix for this implementation. dstPoint target plane coordinates a vector matrix.

```
cv2.findHomography(srcPoints, dstPoints[, method[, ransacReprojThreshold[, mask[, maxIters[, confidence]]]])
```

RANSAC—Each picture pair's homography calculate using a random sample consensus approach. Each RANSAC cycle randomly selects four initial feature matches. Following are the steps for RANSAC: i) select four feature pairs (at random); ii) compute homography H (exact), iii) compute inliers, iv) keep the largest set of inliers, and v) re-compute the least-squares H estimate on all of the inliers. To calculate the homography matrix of images, we need this formula:

$$\sum i(c) = \max_j \|H_i P_j - P_{ij}\|, 1 < j < 4$$

where H_i is the predicted homography at frame i , P_j is the reference frame position of target corner j , and p_{ij} is the manually acquired frame i corner j location.

2.2.4. Matching features

To get the matching features, we use matching techniques. To match every matching descriptor in the Input and query image, we use the descriptor matcher and apply the brute-force hamming approach. We get it from the OpenCV tool. We have many matching algorithms, but we use KNN matching to match every nearest descriptor of the image because KNN produces the best closest matches as compared to any other technique. That makes the best matches for the image. For getting matches, we give the descriptors as input to the matcher. From the Algorithm 1, match-1, and match-2 are the mating points of the input and query images. In our proposed system, we implemented KNN-matching to find the matches for the input images.

Algorithm 1: KNN-matching

- Step 1: Initially, take the descriptors of the input and query the image.
- Step 2: Take the matching ratio (0.8) for testing the values of the image descriptors.
- Step 3: Then we find the Euclidean distance for every input and query image descriptor by the formula.

$$\text{Euclidean Distance: } d_{we}(i, j) = \left(\sum_k^p w_k (X_k^i - X_k^j)^2 \right)^{\frac{1}{2}}$$

- Step 4: After finding the distance, the condition has to satisfy is the distance of the input image descriptor should be less than the matching ratio * distance of the query image i.e., $m.\text{distance} < 0.8 * n.\text{distance}$.
- Step 5: If the condition is satisfied, append the values of the query index of keypoint1 to match-1 and train the index of keypoint2 to match-2.
- Step 6: Match-1 and match-2 are the matching points.
- Step 7: END the process.

2.2.5. Inlier calculation

Inliers are the pure matches of the images calculated inliers of the input and query image. The system needs the homography matrix and the matched keypoints of the pictures. For inlier calculation, we use the homography check and Euclidean distance of the matched keypoints. By the following Algorithm 2, we find the inliers and suitable matches of the images in our proposed work.

Algorithm 2: Inlier calculations

- Step 1: Initially, we take an inlier threshold to identify inliers with a homography check.
- Step 2: Create the homogenous point for matched points.
- Step 3: Then project homogenous points from the input image to the query image.
- Step 4: Calculate the Euclidean distance for the matched keypoints.
- Step 5: Then follow the condition, i.e., a Euclidean distance less than the inlier threshold value.
- Step 6: If step 5 satisfies, find the length of inlier1 and inlier2 with the Dmatch tool of OpenCV and append to suitable matches. Finally, append the input match point to inliers1 and query match point to inliers2.
- Step 7: END the process.

2.2.6. Draw the matches for pure inliers

In our proposed system, we draw the pure inliers to represent the pure matches of the images. To represent the images, we use the OpenCV tool. In OpenCV, we have a pre-defined function CV.draw_matches to draw the matches of the image. Using this model, the proposed system has been implemented to draw the matches of the image. The function represented given:

cv2.drawMatches(image-1, inliers1, image-2, inliers2, and good_matches)

Model:

Image-1: Input image of the system.

Image-2: Query image of the system.

Inliers-1: Input image inliers.

Inliers-2: Query image inliers.

Good_matches: Similar and pure matches of input and query image.

res = np.empty((max(input_image.shape[0], QueryImage.shape[0]), input_image.shape[1] + QueryImage.shape[1], 3), dtype=np.uint8)

Code: cv2.drawMatches(input_image, Inliers_InputImage, QueryImage, Inliers_QueryImage, good_matches, res)

2.2.7. Inlier ratio calculation

The matching produces qualitative outcomes of the image. It is hard to determine whether our detector gives us the best result. To overcome it, we find the inlier ratio to represent our outcome. To represent out, we use the formula to find the inliers ratio. The mathematical formula for inlier ratio:

$$\text{Percentage of Inliers} = 100 \times \frac{\text{Inliers}}{\text{Matches}}$$

3. IMPLEMENTATION

The proposed system was implemented based on the feature descriptors like ORB, LATCH, BEBILD, FREAK, and BRISK. Along with these descriptors KNN matching algorithm and inlier calculations are performed on the feature descriptors. The following steps are used to implement the proposed system.

Implementation of proposed work

- Step 1: The feature descriptors produce descriptors in n-dimensional array to detect the descriptors of the image, as shown in Figures 4 and 5.
- Step 2: Calculate the homography matrix with the feature detectors. As illustrated in Figure 6, we identify the homography matrix to compute inliers.
- Step 3: Finding the matching features of the input and query image. We find the matching features of the image by implementing it as shown in Figure 7.
- Step 4: Calculate the inliers of the matched keypoints with the inlier calculation algorithm. It is the sample implemented to get the inliers for the input and the query image; in inlier calculation, we calculate the good matches of the images for calculating the inliers.

- Step 5: Draw the inliers to the images to represent the pure matches of the image. In our proposed system, we draw the pure inliers to describe the pure matches of the images. To illustrate the images, we use the OpenCV tool.
- Step 6: Calculate the inlier ratio with matches and inliers of the image using the inlier ratio calculation formula. It is implemented in the system to calculate the inlier ratio of the images, as shown in Figure 8.

```
[[ 15 176 251 ... 221 244 118]
 [ 61 232 123 ... 155 196 226]
 [236 125 57 ... 91 198 35]
 ...
 [ 47 252 225 ... 222 210 188]
 [235 222 168 ... 220 244 127]
 [131 58 179 ... 213 39 253]]
```

```
[[ 58 132 110 ... 10 240 34]
 [ 98 16 239 ... 8 55 251]
 [123 123 64 ... 143 11 213]
 ...
 [ 61 240 23 ... 51 6 240]
 [116 249 19 ... 43 43 128]
 [ 56 239 80 ... 158 142 139]]
```

Figure 4. Input image feature descriptors Figure 5. Query image feature descriptors

```
[[ 1.01117223e+00  5.79232024e-02 -7.77440117e+00]
 [ 1.60056844e-02  1.62378309e+00 -2.80059742e+00]
 [ 1.75088436e-04  2.57940521e-04  1.00000000e+00]]
```

Figure 6. The homography matrix for the descriptors

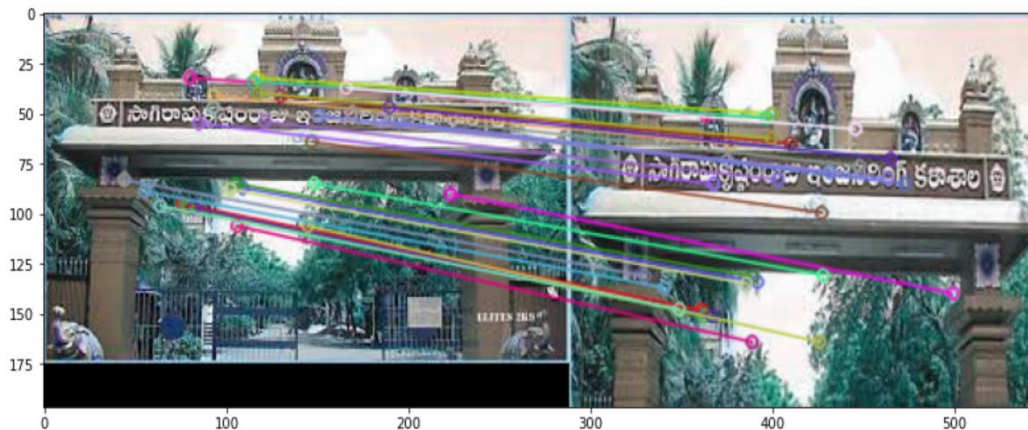


Figure 7. Pure matchings of the input and query images

```
Matching Results
*****
# Keypoints 1:                2145
# Keypoints 2:                1839
# Matches:                    44
# Inliers:                    29
# Inliers Ratio:              65.9090909090909
```

Figure 8. Result of the input and query images

4. RESULTS

The proposed system tested on the different types of images. The image feature extracts using our system with the help of feature detectors to evaluate the image features with the matching features and produce the inlier ratio of the picture. The input and query images show in Figure 9. Scenarios 1-3 are the images considered to get an output as an inlier ratio. Figure 9 consists of three attributes: description, input, query image, and output. Figure 9 shows the results of the evaluated images. In Figures 10 to 12, inlier values used to construct graphs for scenarios 1-3.

Description	Input and Query Image		Output
Real-world images (arch of the college). (Scenario-1)			ORB-35 BRISK-33 BRISF-25 FREAK-25 LATCH-20 BEBLID-52
We are evaluating Real world birthday poster Images. (Scenario-2)			ORB-38 BRISK-48 BRIEF-26 FREAK-30 LATCH-28 BEBILD-54
Evaluating the home curtains of real-world image (Scenario-3)			ORB-34 BRISK-39 BRIEF-32 FREAK-35 LATCH-43 BEBILD-39

Figure 9. Results of the evaluated images

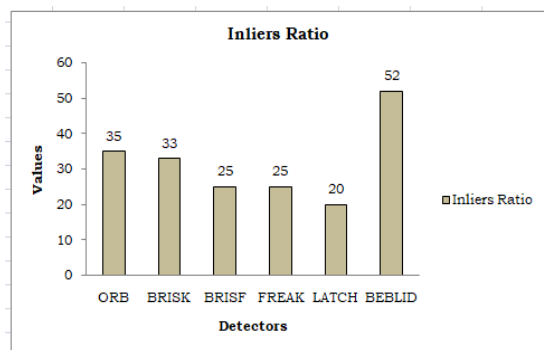


Figure 10. Inlier ratios for the scenario-1

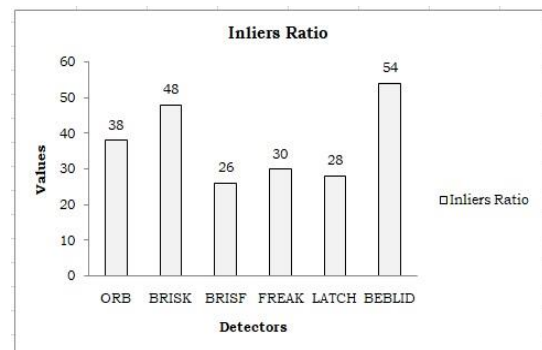


Figure 11. Inlier ratios for the scenario-2

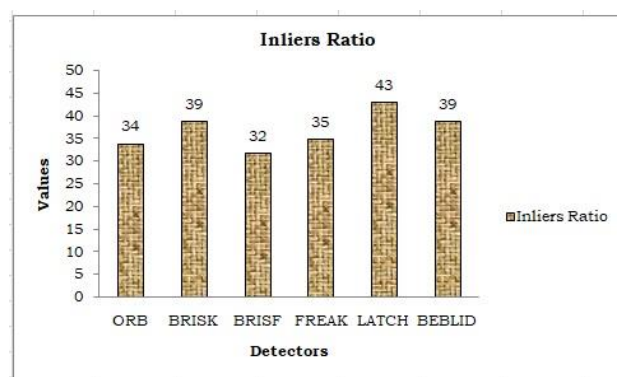


Figure 12. Inlier ratios for the scenario-3

5. CONCLUSION

We use real-world scenarios for evaluating image features using various descriptors to find accurate matches between input and query images. In this, we implement the performance of descriptors on similar images with different angles and other weather conditions. Our proposed system shows which descriptor accurately evaluates features like inlier ratio and matches. Working in the real world gives you complete control over the surroundings and awareness of the scene's geometry. Explore the effects of several environmental factors on descriptor isolation. According to our results, gradient descriptor efficiency for matching keypoints in pictures recorded under different illuminations depends on the spatial structure of the pooling areas. However, Issues with the amount of pooling areas need to be resolved. Images were collected from several camera viewpoints. Due to a lack of distinctiveness, the lower dimensional feature descriptors performed poorly. We ranked inliers ratios about specific image descriptors.




REFERENCES

- [1] M. Calonder, V. Lepetit, M. Ozuysal, T. Trzcinski, C. Strecha, and P. Fua, "BRIEF: Computing a Local Binary Descriptor Very Fast," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 7, pp. 1281–1298, Jul. 2012, doi: 10.1109/TPAMI.2011.222.
- [2] B. Kaneva, A. Torralba, and W. T. Freeman, "Evaluation of image features using a photorealistic virtual world," in *2011 International Conference on Computer Vision*, IEEE, Nov. 2011, pp. 2282–2289, doi: 10.1109/ICCV.2011.6126508.
- [3] B. Igler, "Feature Evaluation for Mobile Applications: A Design Science Approach Based on Evolutionary Software Prototypes," in *International Conference of Design, User Experience, and Usability*, 2013, pp. 673–681, doi: 10.1007/978-3-642-39253-5_75.
- [4] F. Remmen, G. Dubbelman, and P. H. N. de With, "Evaluating image features in real-world scenarios," in *2020 IEEE 5th International Conference on Signal and Image Processing (ICSIP)*, IEEE, Oct. 2020, pp. 535–541, doi: 10.1109/ICSIP49896.2020.9339363.
- [5] D. R. Babu, R. S. Shankar, G. Mahesh, and K. V. S. S. Murthy, "Facial expression recognition using bezier curves with hausdorff distance," in *2017 International Conference on IoT and Application (ICIOT)*, IEEE, May 2017, pp. 1–8, doi: 10.1109/ICIOTA.2017.8073622.
- [6] E. Rosten, R. Porter, and T. Drummond, "Faster and Better: A Machine Learning Approach to Corner Detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 1, pp. 105–119, Jan. 2010, doi: 10.1109/TPAMI.2008.275.
- [7] V. Lepetit and P. Fua, "Keypoint recognition using randomized trees," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 9, pp. 1465–1479, Sep. 2006, doi: 10.1109/TPAMI.2006.188.
- [8] S. A. J. Winder and M. Brown, "Learning Local Image Descriptors," in *2007 IEEE Conference on Computer Vision and Pattern Recognition*, IEEE, Jun. 2007, pp. 1–8, doi: 10.1109/CVPR.2007.382971.
- [9] A. Jakubovic and J. Velagic, "Image Feature Matching and Object Detection Using Brute-Force Matchers," in *2018 International Symposium ELMAR*, IEEE, Sep. 2018, pp. 83–86, doi: 10.23919/ELMAR.2018.8534641.
- [10] H. Porav, T. Bruls, and P. Newman, "I Can See Clearly Now: Image Restoration via De-Raining," in *2019 International Conference on Robotics and Automation (ICRA)*, IEEE, May 2019, pp. 7087–7093, doi: 10.1109/ICRA.2019.8793486.
- [11] L. Sun, K. Wang, K. Yang, and K. Xiang, "See clearer at night: towards robust nighttime semantic segmentation through day-night image conversion," in *Artificial Intelligence and Machine Learning in Defense Applications*, J. Dijk, Ed., SPIE, Sep. 2019, p. 8, doi: 10.1117/12.2532477.
- [12] K. Mikolajczyk and C. Schmid, "A performance evaluation of local descriptors," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 10, pp. 1615–1630, Oct. 2005, doi: 10.1109/TPAMI.2005.188.
- [13] H. Jing, X. He, Q. Han, and X. Niu, "CBRISK: Colored Binary Robust Invariant Scalable Keypoints," *IEICE Transactions on Information and Systems*, vol. E96.D, no. 2, pp. 392–395, 2013, doi: 10.1587/transinf.E96.D.392.
- [14] K. M. Yi, E. Trulls, V. Lepetit, and P. Fua, "LIFT: Learned Invariant Feature Transform," in *European conference on computer vision*, 2016, pp. 467–483, doi: 10.1007/978-3-319-46466-4_28.
- [15] Y. Li, S. Tang, R. Zhang, Y. Zhang, J. Li, and S. Yan, "Asymmetric GAN for Unpaired Image-to-Image Translation," *IEEE Transactions on Image Processing*, vol. 28, no. 12, pp. 5881–5896, Dec. 2019, doi: 10.1109/TIP.2019.2922854.
- [16] G. Hua, M. Brown, and S. Winder, "Discriminant Embedding for Local Image Descriptors," in *2007 IEEE 11th International Conference on Computer Vision*, IEEE, 2007, pp. 1–8, doi: 10.1109/ICCV.2007.4408857.
- [17] S. Winder, G. Hua, and M. Brown, "Picking the best DAISY," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, IEEE, Jun. 2009, pp. 178–185, doi: 10.1109/CVPR.2009.5206839.
- [18] C. Strecha, A. M. Bronstein, M. M. Bronstein, and P. Fua, "LDAHash: Improved Matching with Smaller Descriptors," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 1, pp. 66–78, Jan. 2012, doi: 10.1109/TPAMI.2011.103.
- [19] M. Ozuysal, M. Calonder, V. Lepetit, and P. Fua, "Fast Keypoint Recognition Using Random Ferns," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 3, pp. 448–461, Mar. 2010, doi: 10.1109/TPAMI.2009.23.
- [20] E. Rosten and T. Drummond, "Machine Learning for High-Speed Corner Detection," in *IEEE transactions on pattern analysis and machine intelligence*, 2006, pp. 430–443, doi: 10.1007/11744023_34.
- [21] H. Bay, T. Tuytelaars, and L. Van Gool, "SURF: Speeded Up Robust Features," in *Computer vision and image understanding*, 2006, pp. 404–417, doi: 10.1007/11744023_32.
- [22] M. Agrawal, K. Konolige, and M. R. Blas, "CenSurE: Center Surround Extremas for Realtime Feature Detection and Matching," in *European conference on computer vision*, 2008, pp. 102–115, doi: 10.1007/978-3-540-88693-8_8.
- [23] A. Baumberg, "Reliable feature matching across widely separated views," in *Proceedings IEEE Conference on Computer Vision and Pattern Recognition. CVPR 2000 (Cat. No. PR00662)*, IEEE Comput. Soc., 2000, pp. 774–781, doi: 10.1109/CVPR.2000.855899.
- [24] S. Lowry et al., "Visual Place Recognition: A Survey," *IEEE Transactions on Robotics*, vol. 32, no. 1, pp. 1–19, Feb. 2016, doi: 10.1109/TRO.2015.2496823.
- [25] R. S. Shankar, V. M. Gupta, K. V. S. Murthy, and C. Someswararao, "Object oriented fuzzy filter for noise reduction of Pgm images," in *2012 8th International Conference on Information Science and Digital Content Technology (ICIDT2012)*, Jeju, Korea (South), 2012, pp. 776–782.




- [26] V. M. Gupta, K. Murthy, and R. S. Shankar, "A Novel Approach for Image Denoising and Performance Analysis using SGO and APSO," *Journal of Physics: Conference Series*, vol. 2070, no. 1, p. 012139, Nov. 2021, doi: 10.1088/1742-6596/2070/1/012139.
- [27] R. Szeliski, "Image alignment and stitching: A Tutorial," in *Handbook of mathematical models in computer vision*, 2006, pp. 273–292.
- [28] R. B. Devareddi, R. S. Shankar, K. V. Murthy, and C. Raminaidu, "Image segmentation based on scanned document and hand script counterfeit detection using neural network," *InAIP Conference Proceedings*, vol. 2576, no. 1, p. 050001, 2022, doi: 10.1063/5.0105808.
- [29] A. Alahi, R. Ortiz, and P. Vanderghenst, "FREAK: Fast Retina Keypoint," in *2012 IEEE Conference on Computer Vision and Pattern Recognition*, IEEE, Jun. 2012, pp. 510–517, doi: 10.1109/CVPR.2012.6247715.

BIOGRAPHIES OF AUTHORS






Shiva Shankar Reddy    is an Assistant Professor at the Department of Computer Science and Engineering in Sagi RamaKrishnam Raju Engineering College, Bhimavaram, Andhra Pradesh, India. He is pursuing Ph.D. degree in Computer Science and Engineering with a specialization in medical mining and machine learning. His research areas are image processing, medical mining, machine learning, deep learning, and pattern recognition. He published 30+ papers in International Journals and Conferences. S.S. Reddy has filed 05 patents. His research interests include image processing, medical mining, machine learning, deep learning, and pattern recognition. He can be contacted at email: shiva.csesrkr@gmail.com.






Dr. Veeranki V. R. Maheswara Rao    is a leading Researcher and Academician in Department of Computer Science and Engineering and holds Ph.D. degree. He is currently working as a Professor in the Department of Computer Science and Engineering at Shri Vishnu Engineering College for Women (A), Andhra Pradesh, India. He is actively involved and successfully implemented three projects funded by DST. He has 45 research papers, 17 of which are Scopus-indexed and 7 of which are Web of Science-indexed. He has 23 years of experience that include 6 years of Industry experience, 19 years of Teaching experience and 15 years of research experience. His research interests include data mining, web mining, cloud computing, big data analytics, data science, artificial intelligence, and machine learning. He can be contacted at email: mahesh_vvr@yahoo.com.



Kalidindi Sravani    is Assistant Professor at Sagi Ramakrishnam Raju Engineering College, Department of Computer Science and Engineering, India. She received a B.Tech. degree in Shri Vishnu Engineering College for Women, Department of Computer Science and Engineering, in 2007. She holds an M.Tech. degree in Sagi Ramakrishnam Raju Engineering College, Department of Computer Science and Engineering, in 2014. Her research areas are data mining, machine learning, and IoT. She can be contacted at email: sravani.kalidindi@gmail.com.



Silpa Nrusimhadri    is working as Assistant Professor in the Department of Computer Science and Engineering at Shri Vishnu Engineering College for Women (A), Andhra Pradesh, India. Presently she is pursuing her Ph.D. in Computer Science and Engineering at Centurion University of Technology and Management (CUTM), Odisha, India. She has 13 years of teaching experience and 8 years of research experience. Her research interests include data mining, web mining, big data analytics, text mining, data science, artificial intelligence, and machine learning. She is actively involved and successfully implemented two projects funded by DST. She has 11 research Scopus-indexed papers. She can be contacted at email: nrusimhadri.silpa@gmail.com.