Elevating fingerprint matching with a unified dHash, SIFT, and LSH model

Shreyas Dongre
Artificial Intelligence
MPSTME, NMIMS University
Mumbai, India
shreyas.dongre134@nmims.edu.in

Shrushti Mehta
Artificial Intelligence
MPSTME, NMIMS University
Mumbai, India
shrushti.mehta128@nmims.edu.in

Pallavi Raut
Artificial Intelligence
MPSTME, NMIMS University
Mumbai, India
pallavi.raut105@nmims.edu.in

Dr. Vaishali Kulkarni
Artificial Intelligence
MPSTME, NMIMS University
Mumbai, India
vaishali.kulkarni@nmims.edu.in

2023 IEEE International Conference on Computer Vision and Machine Intelligence (CVMI) | 979-8-3503-0514-2/23/$31.00 ©2023 IEEE | DOI: 10.1109/CVMI59935.2023.10464460

***Abstract*— User authentication is essential for maintaining the security, privacy, and integrity of systems and data. It ensures that only authorized individuals can access specific resources, applications, or sensitive information, preventing potential data breaches, unauthorized modifications, or malicious activities. Biometric identification, particularly fingerprint recognition, has revolutionized authentication. This research paper proposes combining cutting-edge algorithms such as dHash, SIFT-Harris feature points, and Locality-Sensitive hashes to create a more efficient fingerprint-matching system. It aims to improve matching accuracy, data breach avoidance, robustness, and processing speed in fingerprint recognition systems. The final proposed model works for almost all altered (obliteration, central rotation, and z-cut) images. A total of 55,273 images were tested, giving an accuracy of 0.99.**

Keywords— fingerprint matching, image hashing, SIFT keypoints, dhash, LSH fingerprint matching.

I. Introduction

Biometric identification has emerged as a pioneering technology that is revolutionizing the realms of authentication, privacy, and security. With its unparalleled ability to uniquely recognize individuals based on their inherent physiological or behavioral traits, such as fingerprints, iris patterns, voice, or facial features, and its benefits over traditional authentication systems like knowledge-based (e.g., KBA questions, password) or token-based (e.g., ID card) authentication, biometrics offers a compelling alternative [1].

It successfully garners significant attention due to its widespread applications, which include biometrics-based access control systems commonly employed in buildings, offices, and secure areas. On matching the biometric, like the fingerprint or face, of the user to the authorized user's biometric database, access to the user is denied or accepted. Mobile devices, Financial Transactions, Border Security and Immigration Control, Criminal Identification, Identity Verification for Government Services, Forensic investigations, etc. use biometrics as well.

Among the myriad of biometric traits available, Fingerprint recognition, with its distinct and unique ridge patterns, has emerged as a vital biometric identification technique. Some key advantages include the uniqueness of each individual's fingerprints, even among identical twins. Fingerprints also remain relatively stable as compared to other biometric traits throughout a person's life; they are formed during fetal development and retain their pattern until decomposition after death. They are very easy to collect, non-invasive, available to all (except for rare medical conditions), rapid, and very cost-effective. This makes it one of the most feasible, widely adopted, and trusted biometric modalities.
While fingerprint recognition offers numerous benefits, it's essential to consider potential drawbacks, such as concerns about privacy, the possibility of spoofing (fake fingerprints), security concerns, and the need for careful management of biometric databases to ensure data security. Traditional fingerprint recognition systems show a sluggish search rate for matching, which increases processing time and reduces the efficiency of such systems. Proper implementation and adherence to best practices can help mitigate these concerns and maximize the advantages of fingerprint biometrics.

Over the years, researchers have continuously sought innovative approaches to enhance the accuracy, efficiency, and security of fingerprint-matching systems. Primarily, traditional fingerprint recognition systems match the fingerprints collected for authentication with the already collected database of fingerprint images. Normally, the traditional recognition system involves the following main phases: collection, database storage, comparison, and identification or recognition. Since a fingerprint is a mix of ridges and valleys that form its surface, certain unique points or patterns can be identified, for analysis of the fingerprints in a very detailed manner. Yager and Amin [2] recently released a detailed evaluation of fingerprint-matching techniques. Currently, exceptional fingerprint recognition algorithms are computationally quite expensive. Depending on the given vast size of today's databases (e.g. fingerprint database of the FBI which contains more than 173 million fingerprints), there is a constant need for new approaches which can help to lower time and space complexity while querying for a given test sample. To simplify the time and space complexities, fingerprint databases are first divided into buckets which are mutually exclusive. After the classification of the query into one of the buckets, all that remains is to compare it to others in the same bucket. This significantly reduces the number of one-on-one matches and in turn the computational power.

With the advancement of cryptography and other image-processing methods, image hashing and feature extraction have also increased in prominence. The fingerprint image

data is converted to a hash [3], which is a string-like code that not only encrypts the secret fingerprint information but also speeds up matching searches because comparisons are conducted using the hash code rather than actual fingerprint picture samples. Also, keypoints invariant to scale, rotation, and illumination changes can also be identified based on their intensity, contrast, and local structure. These feature points become highly reliable for recognizing and matching objects in different images, even when subjected to varying viewpoints, lighting conditions, or occlusions.

As technology advances, research has shown that the integration of sophisticated algorithms has significantly improved matching accuracy, robustness, and processing speed. Our research paper aims to explore 3 undermentioned algorithms - dHash algorithm, Scale-Invariant Feature Transform (SIFT) Harris feature points, and Locality-Sensitive Hashing (LSH), together to come up with an efficient fingerprint matching system. The algorithms are used together to extract unique features from the images, encode those and store them in the central database for future comparisons.

This paper is organized into five sub-sections. Section II is the literature review that gives a review of the history of fingerprint recognition systems. Our proposed model has been briefly discussed in Section III. The research dataset, performance evaluation metric, and performance of our model have been discussed in Section IV, and Section V concludes the paper along with the possible future scope.

II. Literature Review

Reviewing older literature available in this domain, we could find numerous techniques developed to find a solution to the given problem statement. One of them was a method based on shape context using Scale-Invariant Feature Transform (SIFT) - Harris feature point combined with the minutiae of fingerprint images to incorporate the orientation and descriptor in the minutiae of fingerprint pictures [4], with the goal of extracting the robust minutiae of fingerprint images. Utilizing both their descriptors and the geometric distribution of the robust minutiae points, a short hash vector has been developed for fingerprint storage and matching. Tulyakov et al. (2007) proposes a method for safeguarding fingerprint templates using symmetric hash techniques. Such symmetric functions are employed when the attributes are unordered, as they are in fingerprint minutia. When only a piece of the fingerprint is visible, this procedure is used; pre-alignment of the fingerprints is not required [5]. Kawther & Salah (2018) propose a technique where a single fingerprint is hashed using the MD5 Hash Algorithm to produce a collection of minutiae values. For each fingerprint image, core points (singular points) are found, and the distances between minutiae points and the core points are calculated using these points. By analyzing the separation of various minutiae points from core points in the two fingerprint samples, two fingerprint images may be matched [6]. The above hashing techniques cannot be used directly on images; therefore, observing these shortcomings Perceptual Hashing was used as used in the research by Weng and Preneel (2009). Natural image hashing is accomplished using perceptual hashing approaches.

Accurate feature extraction is the key to this technique. The region-based feature in this article is formed by the angular radial transform, while the contour-based feature is formed via edge detection [7]. The SIFT-based fingerprint retrieval and recognition strategy was proposed by Zhong et al. (2015) which works by assessing the quality of fingerprint features and utilizing a multi-template image feature fusion technique. The level approach of the SIFT matching concept of a near-adjacent priority scale was used to follow progressive matching and speed up matching. The Locality-sensitive hashing (LSH) method was used to execute SIFT storage retrieval with high-dimensional features [8].

We could conclude from the literature review that the combination of dHash values, hashed SIFT keypoints, and Locality-Sensitive Hashing (LSH) for fingerprint matching receives very little attention in the currently available literature. This research gap presents a unique opportunity to bridge the gap between the individual strengths of the three aforementioned features for fingerprint matching that could potentially lead to a more efficient and accurate fingerprint-matching system, with reduced storage requirements and improved scalability for large-scale biometric applications.

III. Proposed Model

The paper emphasizes devising a model that can extract features from raw pixels without depending on traditional fingerprint features and match the fingerprints while improving the speed, precision, and safety of biometric data. The model incorporates three main stages, with minimal pre-processing of fingerprint images throughout. The block diagram shown in Figure 1 summarizes the process of matching. The fingerprint image is first processed using dHash and SIFT algorithm to give a concatenated image signature. Using LSH these signatures are classified according the similar signatures and compared to give a final output.

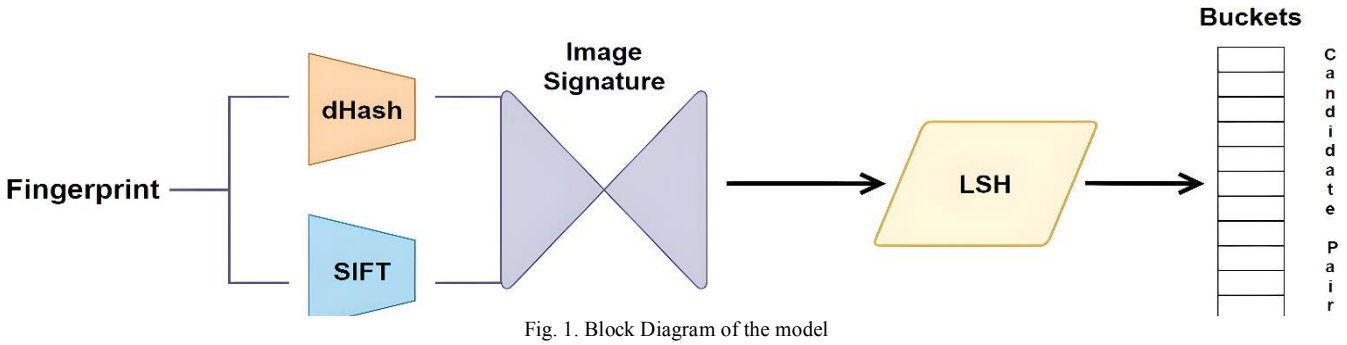

Fig. 1. Block Diagram of the model

A. dHash

Since the last decade, new image-matching algorithms have been developed for various purposes [9, 10]. The most efficient and fastest of them, which are widely used, are Perceptual Hashes [11]. There are different types of image hashing algorithms, but for our application, we are using the dHash algorithm. dHash has been widely used for the detection of copyrighted/illicit images online by matching the computed input test image hash against a database.

We found the algorithm to be very accurate and exceptionally fast. It tracks the relative gradient direction of the image by evaluating the difference between adjacent pixels. Algorithm 1 shows the operations done by the dHash algorithm.

Algorithm 1: dHash Algorithm

INPUT: Fingerprint image of size $r \times w \times c$

OUTPUT: Hash vector [t] of size H_s^2

```
Img = r × w                          ▯Convert image to grayscale
Img = resize(Img)                    ▯Resize the image to size (Hs+1) × Hs
Img = dhash(Img)                     ▯Difference in adjacent pixels. P[x] - P[x+1]
if P[x] > P[x + 1] then              ▯For both, rows and columns
    t .append(1)
else
    t .append(0)
```

The first step generally involves resizing the image, and this is the most time and compute-intensive step. Though our problem domain is computing image similarity, we can not resize the image to a smaller prescribed size, as with fingerprints, we tend to lose the finer details like ridge endings and contours, which play an essential role in determining the general gradient of the image.

To resize an image, we first set a hash-size (H_s) and then resize the image to (H_s+1) × H_s. Hence after thorough experimentation, the most optimal hash-size was found to be 128 bits which results in the image getting resized to 129x128. This leads to around 16,512 pixels. The next step is to reduce the color channels in the image and convert it to grayscale. The dHash algorithm then computes the relative direction of the gradient in the image by computing the difference between the horizontally adjacent pixels. In this case, the 129 pixels per row yield 128 differences between adjacent pixels. One hundred and twenty-eight rows of one hundred and twenty-eight differences become 16,384 bits. Moving forward, we check each pixel in the resultant difference image, by comparing the current pixel with its adjacent pixel (1 if a current pixel has greater color intensity than the adjacent pixel, 0 otherwise), with this, we end up with a 16,384-bit signature of the given fingerprint, representing the light intensity and direction of the gradient. This method works remarkably well in near-duplicate detection because comparing two signatures is a good proxy for comparing the similarity between two images, given that the hyperparameters are well-tuned and it is done correctly.

B. SIFT Keypoints

The SIFT algorithm consists of multiple levels, including keypoint detection, orientation assignment, local descriptor computation, and keypoint matching [8]. This study focuses on keypoint detection and descriptor computation, as these steps are integral to generating a hash for the given fingerprint.

The input image is first loaded for processing. The SIFT detector then gives two main outputs:

1. Points of interests based on the image's scale-space extrema called keypoints, which represent salient image locations invariant to scale changes.
2. For each identified keypoint, their descriptors are extracted, which essentially helps to capture the surrounding visual characteristics, local appearance and gradient information within the region.

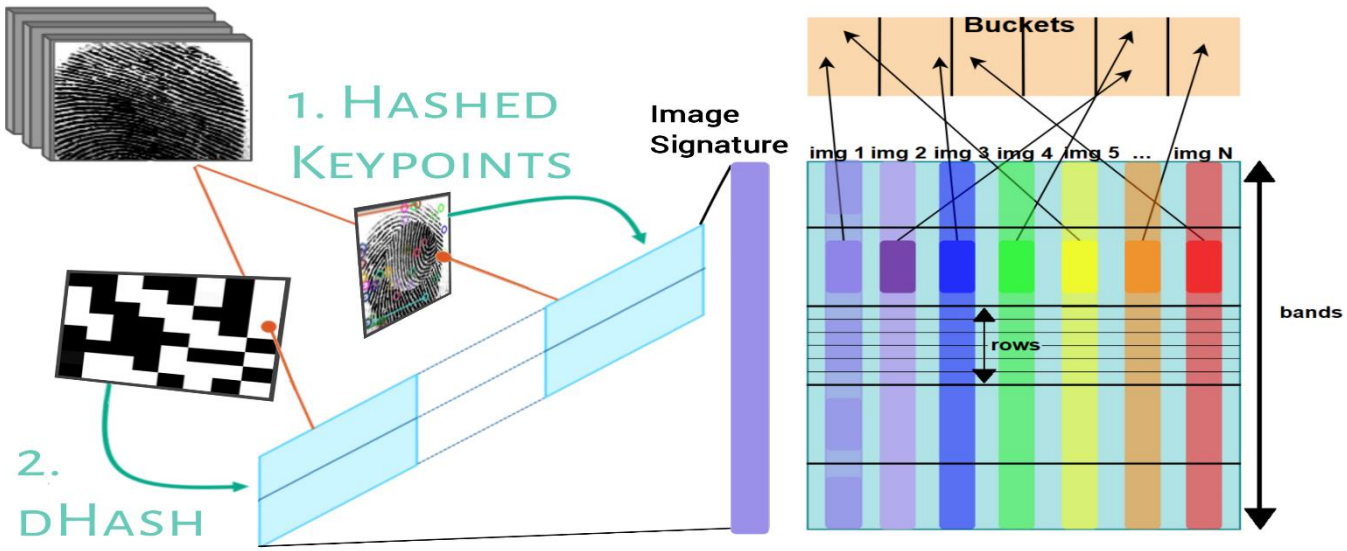

Fig. 2. Architecture of the model

Our encoding algorithm only uses the keypoint coordinates (x,y) that correspond to the detected keypoint locations as

input. The following operations are performed by our encoding algorithm:

- Two Boolean arrays are initialized with all predefined False values, having sizes of 96 and 103 (dimensions of the given images).
- For each coordinate pair, we mark True at indexes that map to the x coordinate in the first array and similarly mark True at positions that map the y coordinate in the second array. This process marks the presence of keypoints at specific locations.
- The two hash vectors of size 96 and 103 are then combined, resulting in a final hash vector of size 199 bits. LSH algorithm is then applied on the hash vectors as shown in Figure 2.

The resulting combined array represents the presence or absence of keypoints at specific locations and acts as a hash value which is unique for every different fingerprint but will be the same for identical fingerprints.

C. LSH

The most crucial part of fingerprint matching and retrieval is implemented by the Locality Sensitive Hashing (LSH) algorithm [12], which is generally used in applications to identify near-duplicate images in a given dataset. In this research, however, we employ LSH to calculate fingerprint signatures, perform bucketing, and compare candidate pairs for similarity.

The hashes obtained from dHash and SIFT algorithm are combined, resulting in one big hash vector of 16,583 bits. This combined hash vector acts as the image signature for a given fingerprint image as shown in figure 2. These image signatures are classified using LSH algorithm.

A dictionary is first initialized to store the packed bits of the image signatures, where each signature is associated with its corresponding fingerprint image file. The image signatures are then partitioned into multiple bands where each band has an equal number of predetermined rows. Iterating through the bands of the hash signature, the algorithm calculates the band's signature for each band and converts it to a byte string. The byte string is then used to index into a hash bucket. If the byte string is not already in the hash bucket, it is initialized as an empty list. The file name of the image is then added to the list. Essentially, the process executed is that, for each band, a hash function is used to calculate a hash value. The hash values are then used to index into hash buckets. Images that hash to the same bucket are likely to be similar.

The resultant hash buckets contain potential matching images together. The algorithm then iterates through the hash buckets and then for each dictionary (hash buckets for a particular band) and identifies pairs of images that are in the same bucket. The pairs of images that are in the same bucket are all added to the set of candidate pairs.

To summarize, instead of directly comparing the image signatures, we divide each signature into a set of number of bands, where each band consists of set number of rows. This implies that the product of number of bands and rows equals hash-size^2, where hash-size^2 represents the length of our signature. Subsequently, we will apply a set number of hash functions, one for each band, to all substrings of signatures within each band. If any of these signature substrings happen to fall into the same bucket for any of the hash functions, we will consider the corresponding signatures and their associated files as candidate pairs, indicating potential near-duplicates. By tuning the rows and bands parameters, we can adjust the sensitivity of candidate pair detection. If we opt for fewer but wider bands, it becomes more challenging for signature substrings (bands) to hash to the same bucket, thus reducing the likelihood of detection. Conversely, selecting more bands and narrower widths increases the probability of signature substrings hashing to the same bucket, making detection easier.

This flavour of LSH implementation was proven highly effective in our experimentation, but it resulted in some false positive cases. Keeping this in mind, another level of checking and verification was included wherein the similarity of images in the candidate pairs was conducted. For each candidate pair, computing the Hamming distance [13] between the unpacked bits of image signatures and calculating the similarity score yielded a better result with decreased false positives.

$$Similarity = \frac{hash\_size^2 - hamming\_dist}{hash\_size^2} \quad (1)$$

In equation (1), $hash\_size$ is the size of the image signature and $hamming\_dist$ is the hamming distance between the two images in a candidate pair. If the similarity is above the set threshold, the algorithm outputs the similarity score, and the matching image is retrieved from the dataset.

IV. Results And Discussion

A. Dataset

Sokoto Coventry Fingerprint Dataset (SOCOFing) dataset prepared by Shehu et al. [14] was used to evaluate the model's performance. The dataset consists of a total of 55,273 images of which 6000 images are real ones consisting of 10 fingerprints one of each finger from 600 subjects. The real images have been altered into 3 more categories of Easy, Medium, and Hard each having 17,934, 17,067, and 14,272 images respectively. The raw resolution of the images were 96 × 103 pixels.

B. Test Build

The research and other processing were carried out on a compute cluster having Nvidia A100 80GB GPU, 128 GB Ram, and Intel Xenon Scalable CPU with 8 virtual cores. All the processes were CPU bound with no external compute required. The implementation was done in Python and executed on Jupyter Notebook.

C. Performance Metric

In evaluating the proposed model's performance, we employed a confusion matrix with four terms: True Positive (TP) which indicates correct positive predictions, False Positive (FP) which denotes incorrect positive predictions, True Negative (TN) which helps in representing correct negative predictions, and False Negative (FN) which indicates incorrect negative predictions. To assess the model's effectiveness, accuracy, precision, recall, and F1 score were computed by using the following formulas.

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad (2)$$

$$Precision = \frac{TP}{TP+FP} \quad (3)$$

$$Recall = \frac{TP}{TP+FN} \quad (4)$$

$$F1\ Score = 2 \times \frac{(\text{Precision} \times \text{Recall})}{(\text{Precision} + \text{Recall})} \quad (5)$$

D. Performance

To test the performance of the model, an identification system was implemented where each fingerprint image was compared against the candidate pairs and the matching mate pair with the highest similarity score was extracted, and subsequently, the extracted pair underwent verification by checking the subject id and finger type to ascertain whether the query and key samples were indeed mate pairs. The following confusion matrix gives us an insight into the model's performance and its strengths.

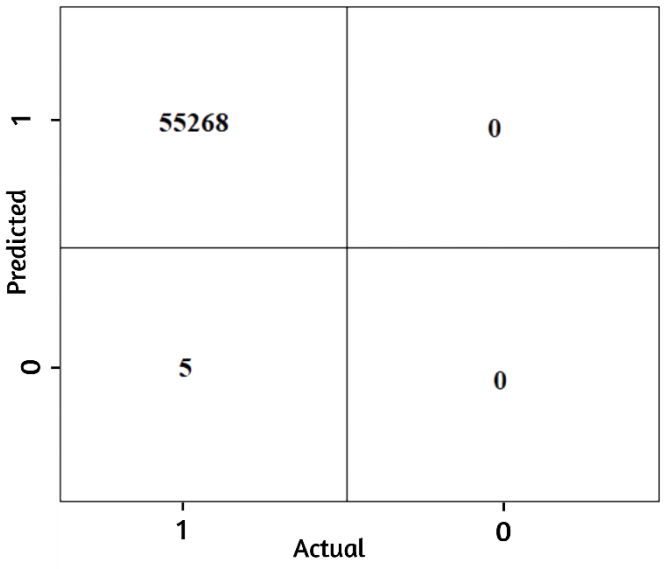

Fig. 3. Confusion Matrix

Table 1 shows the measured performance of the model when tested on the entire dataset.

TABLE I. MODEL PERFORMANCE ON SUB-DATASETS

DATASET	ACCURACY	PRECISION	RECALL	F1
Real	100%	100%	100%	100%
Altered-Easy	100%	100%	100%	100%
Altered-Medium	99.98%	100%	100%	100%
Altered-Hard	99.97%	100%	99.98%	99.99%

On an Nvidia A100 80GB GPU, 128 GB RAM, and Intel Xenon Scalable CPU with 8 virtual cores, it took 50 minutes to complete a 1:N match [15] on each of the 55,273 images. The model's total throughput is around 18 matches per second. Figure 3 and Table 1 together paint a clear picture that, out of 55,273 images, 5 were erroneously accepted as mate pairs. Out of these 5, one was from the Altered-Medium and the remaining four were from the Altered-Hard sub-datasets respectively. If we consider the entire dataset, our model's average accuracy is 99.99%. Our model is robust because it can successfully match the fingerprints from any of the available altered flavors.

Our model fares better as compared to CNN based fingerprint matcher, which was trained on the same dataset [16]. The technique involved extracting the two most important features by PCA and hence training a Convolutional Neural Network. While the CNN approach did achieve a classification accuracy of 99.98% on the Real dataset and 99.94% on Altered Hard. The training time for the model was 1 hour and it was trained only on Real and Altered Hard datasets.

V. CONCLUSION/FUTURE SCOPE

Our work proposes a model that integrates three state-of-the-art algorithms: the dHash algorithm, Scale-Invariant Feature Transform (SIFT) Harris feature points, and Locality-Sensitive hashes to give a final fingerprint match accuracy of 99.99% on SOCOFing dataset. Our experiments showed promising results implying that the model's performance is very promising in terms of throughput and accuracy. Easy feature extraction from fingerprints by dHash, swift keypoint detection by SIFT, and fast matching of LSH help the model perform better as compared to the existing literature. Because the candidate pairs are generated only once during enrolment, the model can be always online, and the fingerprint matching and retrieval can be carried out directly from the list of candidate pairs, leading to a quick result with almost negligible errors.

The most important aspect of the model is user data privacy. During enrolment, the system stores only the dHash and keypoints hashes in the system, meaning no raw fingerprint images or other fingerprint information is ever stored. Hence, even in case of a worst data breach from the system, nobody can ever reverse engineer the hash and reconstruct the fingerprint as there is no information about the image in the hash signatures. The hash signatures data, even if leaked, is useless to any third party. Hence, this approach satisfies all

three aspects of biometric data: improving the speed of matching, having state-of-the-art accuracy, and maintaining user data privacy.

Future scope for the model involves using Speeded Up Robust Features (SURF) instead of SIFT, which is a fast and robust algorithm for local, similarity-invariant representation and comparison of images. SURF is generally faster than SIFT and will be successful in giving higher throughputs while also helping decrease the computational complexity of the algorithm. Similarly, K-D trees and VP trees could be utilized for matching image signatures which could result in reduced search complexity.

REFERENCES

[1] K. Dharavath, F. A. Talukdar and R. H. Laskar, "Study on biometric authentication systems, challenges and future trends: A review," 2013 IEEE International Conference on Computational Intelligence and Computing Research, Enathi, India, 2013, pp. 1-7.

[2] Yager, Neil and Adnan Amin. "Fingerprint classification: a review." *Pattern Analysis and Applications* 7 (2004): 77-93.

[3] Alfiansyah and R. W. Wardhani, "Implementation of Secure Hash Algorithm – 3 for Biometric Fingerprint Access Control Based on Arduino Mega 2560," 2018 International Conference on Applied Information Technology and Innovation (ICAITI), Padang, Indonesia, 2018, pp. 31-35.

[4] R. Muthu, A. Bouridane and F. Khelifi, "Minutiae based fingerprint image hashing," 2014 International Conference on Control, Decision and Information Technologies (CoDIT), Metz, France, 2014, pp. 696-700.

[5] Tulyakov, Sergey & Farooq, Faisal & Mansukhani, Praveer & Govindaraju, Venu. (2007). Symmetric Hash Functions for Secure Fingerprint Biometric Systems. Pattern Recognition Letters. 28. 2427-2436.

[6] A., Kawther & M., Salah. (2018). Authentication using Hashed Fingerprint. International Journal of Computer Applications. 180. 24-27.

[7] L. Weng and B. Preneel, "Shape-based features for image hashing," 2009 IEEE International Conference on Multimedia and Expo, New York, NY, USA, 2009, pp. 1074-1077.

[8] Zhong, Yunfei & Peng, Xiaoqi. (2015). SIFT-Based Low-Quality Fingerprint LSH Retrieval and Recognition Method. International Journal of Signal Processing, Image Processing and Pattern Recognition. 8. 263-272.

[9] Vishal Monga and Brian L. Evans, "Robust perceptual image hashing using feature points," in Proc. of IEEE International Conference on Image Processing, 2004, vol. 1, pp. 677– 680.

[10] Sujoy Roy and Qibin Sun, "Robust hash for detecting and localizing image tampering," in Proc. of International Conference on Image Processing, 2007, vol. 6, pp. 117– 120.

[11] Hamadouche, M., Zebbiche, K., Guerroumi, M., Tebbi, H., Zafoune, Y., 2021.A comparative study of perceptual hashing algorithms: application on fingerprint images. In: 2nd International Conference on Computer Science's Complex Systems and their Application, p. 12. Algeria.

[12] Jafari, Omid & Maurya, Preeti & Nagarkar, Parth & Islam, Khandker & Crushev, Chidambaram. (2021). A Survey on Locality Sensitive Hashing Algorithms and their Applications.Available: https://doi.org/10.48550/arXiv.2102.08942.

[13] "Real Python | Fingerprinting Images for Near-Duplicate Detection." Available: https://realpython.com/fingerprinting-images-for-near-duplicate-detection/#improving-our-algorithm.

[14] Y. I. Shehu, A. Ruiz-Garcia, V. Palade, and A. James, "Sokoto Coventry Fingerprint Dataset". Available :https://arxiv.org/abs/1807.10609.

[15] S. Hemalatha, "A systematic review on Fingerprint based Biometric Authentication System," 2020 International Conference on Emerging Trends in Information Technology and Engineering (ic-ETITE), Vellore, India, 2020, pp. 1-44.

[16] S. H. Mahmood, A. K. Farhan and E. M. El-Kenawy, "A proposed model for fingerprint recognition based on convolutional neural networks," 6th Smart Cities Symposium (SCS 2022), Hybrid Conference, Bahrain, 2022, pp. 343-347, doi: 10.1049/icp.2023.0572.