Empirical Analysis of Feature Points Extraction Techniques for Space Applications

Janhavi H. Borse[1]
Department of Computer Engineering
Smt. Kashibai Navale College of Engineering
Savitribai Phule Pune University, Pune, India

Dipti D. Patil[2]
Department of Information Technology
MKSSS's Cummins College of Engineering for Women
Savitribai Phule Pune University, Pune, India

***Abstract*—Recently, space research advancements have widened the scope of many vision-based techniques. Computer vision techniques with manifold objectives require that valuable features are extracted from input data. This paper attempts to analyze known feature extraction techniques empirically; Scale Invariant Feature Transform (SIFT), Speeded up robust features (SURF), Oriented fast and Rotated Brief (ORB), and Convolutional Neural Network (CNN). A methodology for autonomously extracting features using CNN is analyzed in more detail. The autonomous process demonstrates the use of convolutional neural networks for feature extraction. Those techniques are studied and evaluated empirically on lunar satellite images. For analysis, a dataset containing different affine transformations of a video frame is generated from a sample lunar descent video. The nearest neighbor algorithm is then applied for feature matching. For an unbiased evaluation, a similar process of feature matching is repeated for all the models. Well-known metrics like repeatability and matching scores are employed to validate the studied techniques. The results show that the CNN features showed much better computational efficiency and stable performance concerning matching accuracy for lunar images than other studied algorithms.**

Keywords*—Artificial intelligence; convolutional neural network; computer vision; feature extraction; machine learning; satellite images; space research

I. Introduction

Recent advances in space exploration have opened doors for many research challenges. Processing real-time videos and images captured through spacecraft cameras is one of such challenging tasks. Extracting features useful for further space exploration and navigation tasks is at the primary stage. A spacecraft, once injected into a planet's orbit, keeps on orbiting around the planet. While in orbit, it keeps on capturing videos and images through its onboard cameras. Such kinds of motion result in spatially transformed images of the same scene majority of times. Detecting points of interest from such images or videos in real-time is of paramount importance and a challenging task indeed.

Many proven systems exist which work as image pre-processing techniques for computer vision tasks. There are few areas like image retrieval, medical imaging, object detection, and recognition where these techniques are extensively used. But in this era of automation and artificial intelligence, manual pre-processing of images needs to be avoided. Hence many new systems have been developed with automated feature detection procedures in these domains using deep CNNs. Still, the area of space research has a scope to enter into this automation. This paper intends to analyze a few feature extraction techniques concerning their suitability and sustainability in space applications.

II. Related Work

There are many state-of-the-art algorithms available in the literature as feature detectors & descriptors. Still, their computational complexity does not allow them to be used for real-time tasks. Few comparisons amongst them are available in the reviews [1]–[8]. Many of these algorithms are proposed for detecting and describing points of interest from an image. Initial emphasis was only detecting points of interest or edges from raw images for object detection tasks. Later the focus was shifted to object recognition tasks by taking care of spatial transformations. For keypoint extraction, remarkable work is brought in by the Harris Corner detector [9] and Scale Invariant Feature Transform (SIFT) [10]. Harris Corner detector can extract key points valid for feature tracking algorithms, while SIFT addresses invariance's challenge to affine transformations. But these algorithms were computationally intensive, and hence the next challenge was to speed up the feature detection process. Researchers eventually discovered the new developments like Speeded up robust features (SURF) [11], [12], Features from accelerated segment test (FAST) [13], [14], Binary robust independent elementary features (BRIEF) [15], and Oriented FAST and rotated BRIEF (ORB) [16]. Through the literature, SURF is found to deliver quality features and is computationally efficient as well. ORB, which is a combination of FAST & BRIEF, is computationally speedy than SURF, but the features it extracts are not suitable for image matching tasks. Moreover, these algorithms are standalone versions, and their real-time applicability is questionable. Few works demonstrated the use of feature extraction techniques specific to application domains like medical imaging [17], image retrieval systems [18], and gesture recognition [19], [20].

In the last decades, few deep learning techniques and convolutional neural network techniques[21], [22], [2] are also developed with an abundance in data availability and computationally powerful resources in recent years. The ultimate target of these techniques is image recognition and computer vision task. Many of these techniques rely on already built and tested deep neural network models like Inception [23], VGG [24], XCeption [25], ResNet [26]. Many

researchers have used transfer learning by finetuning ready models for reaching their goals. Two things must be clear in transfer learning before using a particular base model; the first is the dataset on which the model is trained, and the second is the intended application domain. Most of these well-known models are trained on a generalized IMAGENET (http://www.image-net.org) dataset. Hence the knowledge gained through its training is adapted in current research, aiming to deal with data consisting of satellite images and videos.

The real-time requirements to address space research challenges and the existing methods discussed so far motivate us to empirically analyze a few feature extraction techniques like SIFT, SURF, ORB, and CNN. An automated feature extraction process using a convolutional neural network (CNN) is also designed and implemented for experimental analysis. Hence, this work's primary purpose is to study these techniques empirically and analyze their performance concerning time-critical space applications. For this purpose, a dataset consisting of lunar images is constructed from videos captured by a spacecraft's onboard cameras. Each image is spatially transformed with the known transformation matrix. Features are extracted from each image and its transformed versions using all the studied techniques. Image matching using the nearest neighbor algorithm is performed for each image tuple (reference image, transformed image). Ideally, when an image is spatially transformed, its transformed versions show many similarities in detected features as long as a downward-looking camera captures the video with minimal frame delay. One can efficiently compute the ground truth feature vector with known transformations by applying the same transformations to the reference image's features. Finally, their results are validated with available performance metrics and compared with each other.

The details of state-of-the-art techniques like SIFT, SURF, ORB are prevalently available in the literature. Past few years, the scope of CNN is widened due to the automation in the feature extraction process. Still, for few domains like space research, some more analysis is needed for testing its reliability. This paper is intended to perform a comparative analysis of these techniques for space research applicability. For testing the validity of the analysis, CNN features are compared with the SIFT, SURF, and ORB features.

This paper is organized as follows: Section 2 discusses an automated feature extraction process using a CNN architecture. Section 3 elaborates on experimental setup and dataset generation. The performance metrics are discussed in Section 4. Section 5 discusses results and comparisons in classical algorithms, and finally, Section 6 concludes the paper.

III. An Automated Feature Extraction Process using CNN

A. Selecting a ResNet Architecture for CNN Features

Transfer learning is used to generate features. The CNN architecture consists of a ResNet as a base model without a classification layer, as shown in Fig. 1. It is then cascaded with one flatten layer, 2 fully connected (FC) layers at the end. The output of convolutional layers is 3-dimensional maps- $(M \times N \times f)$. The first two dimensions are the size of a feature map, and the last dimension is the number of maps generated at each layer. The number of feature maps corresponds to the number of filters. After the last convolutional layer, a flattened layer is introduced to flatten a 3-dimensional tensor into a single dimension. FC layers at the end serve as an output layer for the Model. The number of computational units in this layer will decide the dimensions of the feature vector. A general deep learning network model expects softmax or other nonlinear functions at the output layer for targeting classification or another more vital task. The goal is to extract features, so the nonlinear function interface at the output layer is removed from the ResNet block.

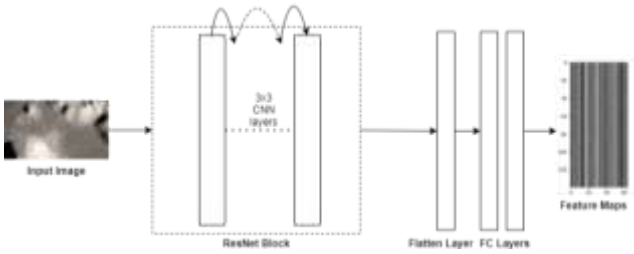

Fig. 1. CNN Model with ResNet base Architecture. Only 64 Features are Shown for Simplicity.

It is assumed that each CNN layer consists of an f-number of $m \times n$ sized high-level convolutional filters. Each input image $I^{i}_{(M,N)}$ is padded to preserve the size of the original image. Then the padded image is passed into the convolution layer to get an output image as,

$$I^{o}_{(M,N)} = I^{p}_{(M+p,N+p)} * f_{(m,n)} \quad (1)$$

In (1), $I^{p}_{(M+p,N+p)}$ does zero-padding form a padded image to an input image $I^{i}_{(M,N)}$ with pad size, p. Rectifier Linear Unit (ReLU) is used as an activation function for both layers. Max pooling is applied to this successive output after padding. A detailed procedure for extracting features through the CNN model is given in a pseudo-code described by algorithm 1.

Using Algorithm 1, CNN features are extracted. The transformed dataset contains both the reference image and its transformed versions. Training is done by applying an 8:2 train-test cross-validation split on the dataset. The details of data collection and creation are described in section 3. The features are extracted from the reference image I^{ref} first and then from its transformed versions: $I^{trans} = (I^{ref}_{R}, I^{ref}_{T}, I^{ref}_{S})$. These features represent the points of interest from each image. Training starts with transfer weights from trained ResNet model. Training is done to minimize the average loss function given by (2). K_p^{gtruth} are the ground truth feature vectors and these are computed using the known transformation parameters. K_p^{trans} are the features extracted from the CNN (Fig. 1(a)). Weights W of the network is adjusted during each epoch to minimize f_{loss} using (2).

$$f_{loss} = \frac{\sum_{D_{trans}} min_W \left\| K_p^{gtruth} - K_p^{trans} \right\|}{|D_{trans}|} \quad (2)$$

Algorithm 1: Procedure for extracting features through a CNN Model.

START

INPUT: Video V

OUTPUT: Features K (1024 x 1)

PROGRAM CNNModel

1. D_{raw} := Call: Function VideoToImage to generate images from video
2. Apply geometric transformations on raw images to generate transformed image dataset,

$$D_{trans} := T_{\text{rotate}}(D_{raw}) \cup T_{\text{scale}}(D_{raw}) \cup T_{\text{translate}}(D_{raw})$$

3. For each image $I^{i}_{(M,N)}$ in D_{trans} REPEAT:

$$I^{p}_{(M+p,N+p)} := Padding\left(I^{i}_{(M,N)}\right)$$

For all filters in a Layer REPEAT:

$$I^{o}_{j(M,N)} := I^{p}_{(M+p,N+p)} * f^{j}_{(m,n)}$$

$$I^{R}_{j(M,N)} := ReLU\left(I^{o}_{j(M,N)}\right)$$

$$I^{PR}_{j(M+p,N+p)} := Padding\left(I^{R}_{j(M,N)}\right)$$

$$I^{Pool}_{j(M,N)} := MaxPool\left(I^{PR}_{j(M+p,N+p)}\right)$$

END REPEAT

1. REPEAT step 3 for next CNN Layer
2. Flatten each image into a single-dimensional vector,

$$V^{i}_{(MxNxf,1)} := Flatten\left(I^{Pool}_{j(M,N)}\right)$$

3. Pass this vector through FC layer,

$$K^{i}_{(256,1)} := FC\left(V^{i}_{(MMxNxf,1)}\right)$$

END REPEAT

END

After training the network, it is evaluated on the rest of the test images. The working of the first algorithm is as follows. Initially, images were extracted from a lunar video to generate a raw image dataset D_{raw} . Then, the following affine transformations were applied to produce a transformed image dataset D_{trans}.

$$M_{scale} = \begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$M_{translate} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix}$$

$$M_{rotate} = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (3)$$

A translation factor of (t_x, t_y) pixels, a rotational angle of $\theta^0$and a scale factor of (s_x, s_y) were used as described by (3). Each image from this new dataset is fed to CNN. Image dimensions are 640×360. Each image is padded to preserve the dimensions and then convolved with f filters of size 3×3. Rectifier Linear Unit (ReLU) is the nonlinear activation function applied to the convolved output. It generates feature maps of size $640 \times 360 \times f$. It is downsampled by using a max-pooling operation. The process is repeated for each next layer with $2 * f$ filters of size 3×3 till the last layer. In the end, the 3-dimensional feature maps are converted into a 1-dimensional tensor using a flatten layer. These flattened feature maps were used as feature vectors of the CNN model.

Algorithm 2 is implemented to track similar feature points from the reference image and transformed image for evaluating the feature similarity. Initially, for each image in the dataset, the features are extracted from all four known techniques, SIFT, SURF, ORB, and CNN, to extract feature vectors K_p^{ref} *and* K_p^{trans} . Then these feature vectors are passed on to the similarity matching using the nearest neighbor algorithm. Scores of matchings between the two image features are used to evaluate different techniques under consideration.

Algorithm 2: Extracting Similar Feature Points from Reference Image and Transformed Image

START

INPUT: Images: Reference I^{ref} and transformed image I^{trans}

OUTPUT: Matched keypoints $K_p^{matched}$

PROGRAM TRACKeypoints

For each image pair (I^{ref}, I^{trans}) in D_{trans} DO:

1. Apply any of the feature detectors like SIFT, SURF, ORB, and CNNModel to get extract keypoints K_p^{ref} and K_p^{trans}
2. Apply image matching technique to find matched key points,

$$K_p^{matched} := NearestNeighbour\left(K_p^{ref}, K_p^{trans}\right)$$

3. Pass vector $K_p^{matched}$ to further compute repeatability and matching scores between two images.

END

IV. Experimental Setup and Dataset Generation

The CNN model was implemented on a 2.4 GHz Intel Core i7 processor with 16 GB DDR4 RAM. Code scripts were written in Python 3.7 with tensor flow framework as backend.

For this research, a python script was written for generating images from a spacecraft landing video. The video is publicly available on the website, https://svs.gsfc.nasa.gov/. This video is an animated view of the landing site of Apollo 17 - Lee Lincoln scarp. The sources created this visualization from Lunar Reconnaissance Orbiter (LRO) photographs and elevation mapping. The video's frame rate is found to be 25 fps, and hence each 25th image frame was captured and stored as an image. In all, 915 grayscale images were generated, which forms the raw image dataset. In the raw image dataset, 200 images were selected at random, and geometric

transformations are applied to create the transformed image dataset containing 800 images. The size of each image in the dataset is 640x360 pixels. Out of the whole dataset, 640 images were used for training, and the remaining 160 are used for testing. Sample images from the dataset are shown in Fig. 2. The complete procedure for dataset generation and description can be found in our prior work [22].

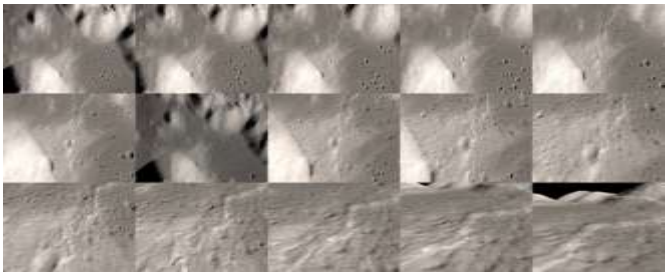

Fig. 2. Montage of Sample Images from the Raw Image Dataset.

The performance of the implemented CNN is evaluated using the metrics repeatability score and matching score [3], [6], [27]–[29], which are widely used for the evaluation of feature detectors. For validating the results of CNN, known feature detectors like SIFT, SURF, and ORB are implemented in the python and Opencv environments. SIFT Lowe's implementation [10] was directly used with few modifications. The ORB algorithm is implemented with two different variations corresponding to the number of feature points extracted equally to 1000 and 500.

The following procedure is followed for computing these metrics to perform an unbiased evaluation of studied feature detectors. Initially, features are extracted from the reference image and then from its transformed version. The descriptors of both images were passed to a Fast Library for Approximate Nearest Neighbours (FLANN) matching algorithm to find a matched point from a transformed image similar to the reference image. This algorithm tends to see the overlapped region from both the images and then returns all the points that match the points in the reference image using a known homography. All the key points in this common region are called correspondences between the two images. After computing the maximum correspondences, the algorithm tries to find the correct matches using some threshold. These interim computations help to calculate the scores of performance parameters.

V. Performance Metrics

For evaluating and comparing the performances of the feature detectors, performance metrics, namely repeatability, matching score, and time taken for feature extraction, are employed.

Image correspondences are key points in common regions between two images using known homography. Repeatability is the measure of the robustness of a feature detector to the external image transformations. The repeatability score is calculated by using (4).

$$\text{Repeatability} = \frac{C^+}{F_{ref}} \tag{4}$$

In (4), C^+ is maximum image correspondences, F_{ref} is the number of features of the reference image.

A matching score measures accuracy while matching the descriptors of logically the same key points from two different images. The matching score is calculated by the formula given by (5).

$$Matching\ Score = \frac{C^+ \cap C^*}{F_{ref}} \tag{5}$$

As in (4), C^* is a number of correct matches.

VI. Results and Discussion

A. Performance Evaluation using Repeatability

The automated feature extraction process results using CNN are compared with conventional methods like SIFT, SURF, and ORB. The distribution of repeatability scores on a percentage scale for all the studied algorithms is shown in Fig. 3. Both versions of SIFT are having almost similar distributions of repeatability ranging between an interval [40,100]. SURF values are found to lie within a range of [55,100], while the range for ORB is [40,100]. ORB features seem to be more minor variants to the transformations, while SIFT & SURF features are highly variant to the input transformations. Graphs show that the CNN model for all the test samples has retained the constant repeatability score of 100%. It means that CNN features are more efficient in finding repeated regions of interest. In the transformations like rotation and scaling, most image regions are repeated. But during a translational shift, few new image regions are added while few regions are subtracted from the original image. An ideal feature detector must take such changes into account. But CNN has neglected the translational shift. The extracted descriptors for a common region of the transformed image always find a proper match for reference descriptors with minimum losses. Most of the features are matched between the two images. As the same procedure is followed for finding key point matches between the two images, it can be concluded that the CNN features are more robust to external transformations than the other classic methods. As it shows no variation in the repeatability scores, it is one of the stable feature detectors. When tuned to generate a more significant number of features (ORB1000), it results in more variations in repeatability than ORB500 and hence can generalize better.

Fig. 4 shows the average repeatability score obtained by all the studied techniques. The score is high for ORB-500 and CNN64. For ORB-1000 also is comparable. But for SURF and SIFT, it is less than 90%. Overall, ORB and CNN are found more robust and hence showed stable performance in terms of repeatability of features. SIFT and SURF are quite unstable as far as this dataset is concerned. CNN features that showed the highest average repeatability show that it is invariant to the external factors such as camera position, angle, motion while extracting the image features. Such transformations are widespread in real-time captured data and hence need a stable feature detector. Thus, CNN model can generalize better if tuned for a more significant number of output features and trained on more data samples.

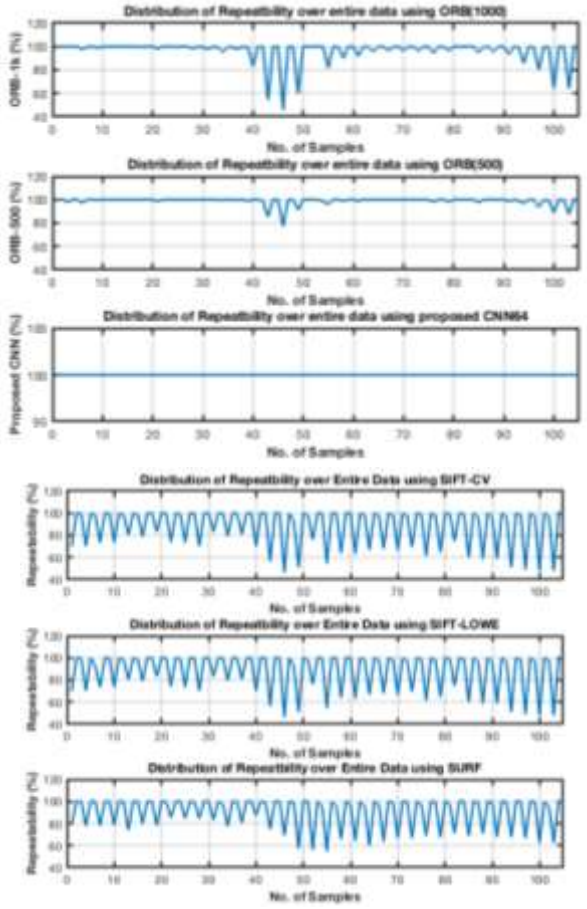

Fig. 3. Distribution of Repeatability Score obtained from Studied Techniques.

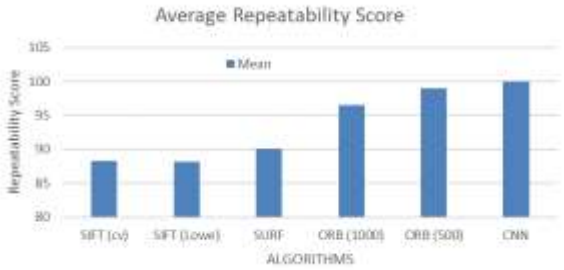

Fig. 4. Average Repeatability Score obtained from Various Algorithms.

B. Performance Evaluation using Matching Score

In addition to repeatability, a matching score is yet another evaluation parameter employed to quantify the performance of the studied techniques. Fig. 5 shows a box plot for matching scores from the features obtained through CNN and conventional algorithms. SIFT and ORB-generated features show skewness in the results, which does not seem stable and reliable data for image recognition or classification tasks. The skewness in the data might bias the model for the following tasks. SURF seems reasonably reliable but has introduced greater variance to changing inputs. CNN boxplot is concentrated near the mean value, and it does not show any skewness in the results. Hence CNN based model seems unbiased, and hence more stable.

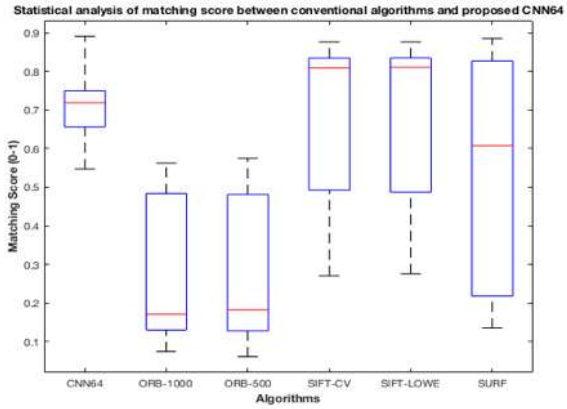

Fig. 5. Statistical Analysis of Matching Score for different Algorithms.

Table I shows the values of computed statistics. The average matching score obtained through ORB is the least of all. The highest score is obtained through CNN. Although the average matching score obtained through SIFT is not far less than that obtained through CNN, SIFT generated score shows the more significant variance in computation. Ideally, the matching score should be on a higher side with minor variance as the images are part of a shorter duration video, mainly capturing the same ground scene.

TABLE I. PRIMARY STATISTICS OF MATCHING SCORE OBTAINED FROM DIFFERENT ALGORITHMS

Algorithms/Stats	SIFT-cv	SIFT-Lowe	SURF	ORB-1000	ORB-500	CNN
Mean	0.6917	0.6893	0.5495	0.2655	0.2696	0.7107
Median	0.8090	0.8101	0.6079	0.1713	0.1826	0.7187
Variance	0.0367	0.0369	0.0716	0.0290	0.0294	0.0037

On the other hand, SURF and ORB have shown significantly lesser matching scores than the CNN model. ORB shows consistency in the matching score computation with minor variation compared to SIFT and SURF, but it is more significant than CNN. Once again, the CNN model has shown invariance against the transformations. Overall, concerning matching scores, the CNN features have shown better performance than others.

To measure consistent and robust performance, we run a one-way Analysis of Variance (ANOVA) test to prove our hypothesis for matching scores computed through the application of all the algorithms. For testing the hypothesis, we selected 100 random samples out of the whole dataset with replacement. We run the ANOVA test for 10 such samples. We assumed that a stable and robust detector would always show negligible between-group variance, and hence its sample means are more equivalent to the grand mean of the population. We rigorously tested each sample mean against its grand mean for each algorithm listed in Table I. We run the test with a 95% of a confidence interval. For CNN, we found our assumption held throughout all samples. The assumption did not hold in the case of other algorithms. Few sample means were far away from the grand mean as in Table I. It proved the robustness and consistent behavior of the CNN compared to other feature detectors.

C. Performance Evaluation using Computational Time

The most critical evaluation parameter that needs to take care of for real-time data is the computation time required to detect and extract points of interest. Fig. 6 shows the joint bar graph, which describes the time needed for processing each image by the proposed CNN model and its companion algorithms. The red bar indicates the number of descriptors detected by an algorithm, and the blue bar shows the time required to perform that task.

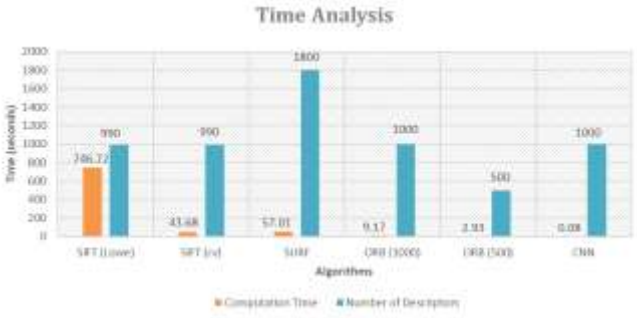

Fig. 6. Algorithmic Analysis of Average (Per Image) Processing Time.

The bar graph of Fig. 6 shows the number of descriptors extracted by each algorithm along with computation time. CNN shows the shortest average processing time amongst all. Both ORB versions show the following smallest time requirement. Then comes SURF, SIFT(cv), and at last SIFT(Lowe). SIFT Lowe's version consumes the highest time compared to others because it processes 2-dimensional data without pre-processing it to any time-efficient form. SIFT and SURF attempt to extract all possible key points from an image and lag in time performance. During tensor flow implementation of CNN, each image is converted into the most efficient tensor representation and then processed further; hence the time required for feature extraction gets drastically reduced. Overall, the evaluation of the CNN is better than the listed algorithms for generated lunar image data.

VII. Conclusion

In this paper, state-of-the-art algorithms for feature extraction are implemented and analyzed on lunar descent image data in detail. A similar process is followed for unbiased evaluation, and known metrics of repeatability, matching score accuracy, and extraction time are used to compare implemented algorithms.

From the detailed analysis of results, it is observed that the CNN model has outperformed the studied conventional algorithms based on suggested performance metrics. The CNN model is capable of handling real-time data with less time requirement. Once a few network parameters are decided, CNN does its job automatically using the input data. No hand-crafted tasks such as image pre-processing, image localization, segmentation are needed as in conventional algorithms.

In effect, the overall performance of the CNN architecture, when compared to existing algorithms, showed much better computational efficiency and stability. The analysis shows that CNN's more profound architecture with transfer learning can be used to meet the real-time demands of space research. But, vigorous training and validation using extensive data are necessary to generalize the model to a greater extent. Extension to work is validating the model by using generated features for object detection tasks.

Acknowledgment

Firstly, we are grateful to Dr. Vinod Kumar, Division Head, U R Rao Satellite Centre ISRO, India, for advising and guiding us in understanding space-related issues.

We want to thank ISRO, India, for supporting & funding this research. We want to express special thanks to Smt. Kashibai Navale College of Engineering and Savitribai Phule Pune University India, for providing us opportunity and support.

References

[1] Ali Ismail Awad and Mahmoud Hassaballah, Image Feature Detectors and Descriptors, vol. 630, no. February. 2016.

[2] P. Fischer, A. Dosovitskiy, and T. Brox, “Descriptor Matching with Convolutional Neural Networks: a Comparison to SIFT,” pp. 1–10, 2014, [Online]. Available: http://arxiv.org/abs/1405.5769.

[3] B. Istanbul, “Analysis of Feature Detector and Descriptor Combinations with a Localization Experiment for Various Performance Metrics Ertugrul BAYRAKTAR* , Pınar BOYRAZ.”

[4] K. Lenc and A. Vedaldi, “Large scale evaluation of local image feature detectors on homography datasets,” Br. Mach. Vis. Conf. 2018, BMVC 2018, 2019.

[5] S. Li, “A review of feature detection and match algorithms for localization and mapping,” IOP Conf. Ser. Mater. Sci. Eng., vol. 231, no. 1, 2017, doi: 10.1088/1757-899X/231/1/012003.

[6] K. Mikolajczyk et al., “A comparison of affine region detectors,” Int. J. Comput. Vis., vol. 65, no. 1–2, pp. 43–72, 2005, doi: 10.1007/s11263-005-3848-x.

[7] G. M. Moura and R. L. D. S. Da Silva, “Analysis and evaluation of feature detection and tracking techniques using OpenCV with focus on markerless augmented reality applications,” J. Mob. Multimed., vol. 12, no. 3–4, pp. 291–302, 2017.

[8] E. Salahat and M. Qasaimeh, “Recent advances in features extraction and description algorithms: A comprehensive survey,” Proc. IEEE Int. Conf. Ind. Technol., pp. 1059–1063, 2017, doi: 10.1109/ICIT.2017.7915508.

[9] M. Harris, C. and Stephens, “A Combined Corner and Edge Detector,” in In C. J. Taylor, editors, Proceedings of the Alvey Vision Conference, 1988, pp. 23.1-23.6, doi: 10.5244/C.2.23.

[10] D. G. Low, “Distinctive image features from scale-invariant keypoints,” Int. J. Comput. Vis., pp. 91–110, 2004, [Online]. Available: https://www.cs.ubc.ca/~lowe/papers/ijcv04.pdf.

[11] H. Bay, T. Tuytelaars, and L. Van Gool, “LNCS 3951 - SURF: Speeded Up Robust Features,” Comput. Vision–ECCV 2006, pp. 404–417, 2006, [Online]. Available: http://link.springer.com/chapter/10.1007/11744023_32.

[12] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool, “Speeded-Up Robust Features (SURF),” Comput. Vis. Image Underst., vol. 110, no. 3, pp. 346–359, 2008, doi: 10.1016/j.cviu.2007.09.014.

[13] E. Rosten and T. Drummond, “Machine Learning for High-Speed Corner Detection,” pp. 430–443, 2006.

[14] E. Rosten, R. Porter, and T. Drummond, “Faster and better: A machine learning approach to corner detection,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 32, no. 1, pp. 105–119, 2010, doi: 10.1109/TPAMI.2008.275.

[15] S. Leutenegger, M. Chli, and R. Y. Siegwart, “BRISK: Binary Robust invariant scalable keypoints,” Proc. IEEE Int. Conf. Comput. Vis., no. November, pp. 2548–2555, 2011, doi: 10.1109/ICCV.2011.6126542.

[16] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "ORB: An efficient alternative to SIFT or SURF," Proc. IEEE Int. Conf. Comput. Vis., pp. 2564–2571, 2011, doi: 10.1109/ICCV.2011.6126544.

[17] A. Reema Matthew, A. Prasad, and P. Babu Anto, "A review on feature extraction techniques for tumor detection and classification from brain MRI," 2017 Int. Conf. Intell. Comput. Instrum. Control Technol. ICICICT 2017, vol. 2018-January, pp. 1766–1771, 2018, doi: 10.1109/ICICICT1.2017.8342838.

[18] S. Dhingra and P. Bansal, "Experimental analogy of different texture feature extraction techniques in image retrieval systems," Multimed. Tools Appl., vol. 79, no. 37–38, pp. 27391–27406, 2020, doi: 10.1007/s11042-020-09317-3.

[19] A. Sharma, A. Mittal, S. Singh, and V. Awatramani, "Hand Gesture Recognition using Image Processing and Feature Extraction Techniques," Procedia Comput. Sci., vol. 173, no. 2019, pp. 181–190, 2020, doi: 10.1016/j.procs.2020.06.022.

[20] H. Dino et al., "Facial Expression ssRecognition based on Hybrid Feature Extraction Techniques with Different Classifiers," TEST Eng. Manag., vol. 83, no. 22319, pp. 22319–22329, 2020.

[21] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in International Conference on Medical image computing and computer-assisted intervention, 2015, pp. 234–241.

[22] J. H. Borse, D. D. Patil, and V. Kumar, "Tracking Keypoints from Consecutive Video Frames Using CNN Features for Space Applications," Teh. Glas., vol. 15, no. 1, pp. 11–17, Mar. 2021, doi: 10.31803/tg-20210204161210.

[23] C. Szegedy et al., "Going deeper with convolutions," Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit., vol. 07-12-June, pp. 1–9, 2015, doi: 10.1109/CVPR.2015.7298594.

[24] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 3rd Int. Conf. Learn. Represent. ICLR 2015 - Conf. Track Proc., pp. 1–14, 2015.

[25] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," Proc. - 30th IEEE Conf. Comput. Vis. Pattern Recognition, CVPR 2017, vol. 2017-Janua, pp. 1800–1807, 2017, doi: 10.1109/CVPR.2017.195.

[26] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit., vol. 2016-Decem, pp. 770–778, 2016, doi: 10.1109/CVPR.2016.90.

[27] S. Ehsan, N. Kanwal, A. F. Clark, and K. D. McDonald-Maier, "Improved repeatability measures for evaluating performance of feature detectors," Electron. Lett., vol. 46, no. 14, pp. 998–1000, 2010, doi: 10.1049/el.2010.1442.

[28] T. Mouats, N. Aouf, D. Nam, and S. Vidas, Performance Evaluation of Feature Detectors and Descriptors Beyond the Visible, vol. 92, no. 1. Journal of Intelligent & Robotic Systems, 2018.

[29] J. L. Sch¨onberger, H. Hardmeier, T. Sattler, and M. Pollefeys, "Comparative Evaluation of Hand-Crafted and Learned Local Features," pp. 01–10, 2017.