1. Introduction

Bad weather conditions such as haziness, mist, foggy and smoky degradation in the quality of the outdoor scene. It is an annoying problem to photographers as it changes the colors and reduces the contrast of daily photos, it diminishes the visibility of the scenes and it is a threat to the reliability of many applications like outdoor surveillance, object detection, it also decreases the clarity of the satellite images and underwater images. So, removing haze from images is an imperative and broadly demanded area in image processing. The large quantities of these suspended particles in the atmosphere cause scattering of light before it reaches the camera which corrupts the outdoor image quality. Haze attenuates the reflected light from the scenes and blends it with additive light in the atmosphere. Haze removal techniques tend to improve this reflected light (i.e., scene colors) from mixed light. The constancy and strength of the visual system can also be improved by using this effective haze removal of image. There are many methods available to remove haze from images like polarization, independent component analysis, dark channel prior etc. Image dehazing technique is essentially a way to minimize or even eliminate interferences through special techniques, to produce adequate visual effects and gain more valuable information. It is always challenging to remove the artefacts and noise from the dehaze image.

1.1. Theoretical Background

An image may be defined as a two-dimensional function, f(x, y), where x and y are spatial coordinates, and the amplitude of f at any pair of coordinates (x, y) is called the intensity or gray level of the image at the point. When x, y, and the amplitude values of f are all finite, discrete quantities, we call the image a digital image. The field of digital image processing refers to processing digital images by means of digital computers. Digital image is composed of a finite number of elements, each of which has a particular location and value. These elements are referred to as picture elements, image elements, pels, and pixels. Pixel is the term most widely used to denote the elements of a digital image. Sometimes a distinction is made by defining image processing as a discipline in which both the input and output of a process are images. Filters are one of digital image enhancement techniques used to sharpen the image and to reduce the noise in the image. There are two types of enhancement techniques called

Spatial domain and Frequency domain techniques which are categorized again for smoothing and sharpening the images.

Haze causes problems in various computer vision and image processing-based applications as it diminishes the scene's visibility. The air light and attenuation are two main phenomena responsible for haze formation. The air light enhances the whiteness in the scene and contrast gets reduced by attenuation. Haze removal techniques help in recovering the contrast and color of the scene. These techniques have found many applications in the area of image processing such as consumer electronics, object detection, outdoor surveillance etc.

1.2. Aim of the proposed Work:

To implement dehazing of images algorithm and make an android app for the same.

1.3. Objective(s) of the proposed work

- To implement a dehazing algorithm using JAVA.
- To make an Android app for clicking or choosing pictures from the gallery.
- To process the images from the app using our dehazing java code.

2. Literature Survey

2.1. Survey of the Existing Models/Work

TITLE	AUTHOR	JOURNAL AND DATE	KEY CONCEPTS
A Survey of Image Enhancement Techniques	Rupneet Kaur Hanspal, Kishor Sahoo	International Journal of Science and Research (IJSR). Issued May 2017	Usage of recent Image Enhancement Techniques to give an idea of each technique’s pros and cons, and

			which one is significant in which field
A Review on Image Contrast Enhancement Techniques	Pooja Patel, Arpana Bhandari	IJO-SCIENCE (International Journal Online of Science) Issued July 2019	Different Image Contrasting Techniques are performed to enhance an Image using different techniques and algorithms
Deep photo cropper and enhancer	Aaron Ott, Amir Mazaheri, Neils Lobo, Mubarak Shah	CRCV Florida Issued 3 August 2020	Hough transformation and color histogram normalization.
Histogram Based Image Enhancement Techniques: A Survey	P. Gupta, J.S. Kumare, U.P. Singh, R.K. Singh	International Journal of Computer Science and Engineering Issued June 2017	Using various Histogram Based Image Enhancement Techniques, enhance an Image without degrading the quality of an image.

Image Enhancement and object Recognition for night vision surveillance	Aashish Bhandari, Aayush Kafle, Pranjal Dhakal, Prateek Raj Joshi, Dinesh Baniya Kshatri	IOE, Tribhuvan University Issued 10 June 2020	Active illumination, FPGA, NIR Pi-Calibration
An Investigation in Satellite Images Based on Image Enhancement Techniques	R. Ablin, C. Helen Sulochana, G.Prabin	European Journal of Remote Sensing Published 02 Oct 2019	A survey on various Satellite Image Enhancement Techniques which recommends fusion-based enhancement while comparing with non-fusion-based enhancement techniques.
Exploring Image enhancement for Salient object Detection in low light Images	Xin Xu, Shiqin Wang, Zheng Wang, Ruimin Hu	ACM Trans.multimedia Comput.commun Vol.1 Issued 31 July 2020	Non-local-block Layer, pixel-level human-labeled ground truth annotations

A Conceptual Study on Image Enhancement Techniques for Fingerprint Images	K. Krishna Prasad, P.S. Aithal	International Journal of Applied Engineering and Management Letters (IJAEML) Issued June 2017	Discussing approaches and methods for reducing noise or impurities and to improve the quality of the image before matching them. The techniques help the fingerprint recognition system to become robust and to obtain high quality in the matching process.
Dark Region-Aware Low Light image enhancement	Junseok Kwon, Dokyeong Kwon, Guisik Kim	EESS.IV Issued 28 August 2020	Dark region-aware low-light image enhancement (DALE),a method to preserve the color, tone, and brightness of original images and prevent normally illuminated areas of the images from being saturated and distorted.

NOISE-AWARE TEXTURE-PRESERVING LOW-LIGHT ENHANCEMENT	Zohreh Azizi, Xuejing Lei, C.-C. Jay Kuo	Media Communications Lab University of Southern California Issued 2 September 2020	Low-light enhancement, · retinex model, denoising.
A remote-sensing image enhancement algorithm based on patch-wise dark channel prior and histogram equalisation with colour correction.	Fayaz Ali Dharejo, Yuanchun Zhou, Farah Deeba, Munsif Ali Jatoi, Yi Du1, Xuezhi Wang	IET Wiley, Image Process. 2021	DCP technology is applied to estimate the haze density through a section of suitable patch size. We used the histogram equalisation technique to this method in order to test the robustness in comparison with other methods. Finally, the colour correction technique, PWLT, is applied to the final output.
A low-light image enhancement method based on bright channel prior and maximum colour channel	Ghada Sandoub, Randa Atta, Hesham Arafat Ali, Rabab Farouk Abdel Kader	IET Wiley, Image Process. 2021	The proposed algorithm consists of two main stages, namely, the fusion-based bright channel estimation and the refinement.

Low-Light Image Enhancement Using Volume-Based Subspace Analysis	Wonjun Kim, Ryong Lee, Minwoo Park, Sang-Hwan Lee, Myung-Seok Choi	IEEE Access, 2020	The proposed solution exploits the volume, which is constructed by stacking a group of image patches, for accurately estimating the illumination component via the corresponding subspace analysis.
Classification-Driven Dynamic Image Enhancement	Vivek Sharma, Ali Diba, Davy Neven, Michael S. Brown, Luc Van Gool, Rainer Stiefelhagen	IEEE Explore, 2019	The paper proposes a unified CNN architecture that can emulate a range of enhancement filters with the overall goal to improve image classification in an end-to-end learning approach.
DeepLPF: Deep Local Parametric Filters for Image Enhancement	Sean Moran, Pierre Marza, Steven McDonagh, Sarah Parisot, Gregory Slabaugh	IEEE Explore, 2020	Automated parameterization of filters for spatially localized image enhancement.

2.2. Summary/Gaps identified in the Survey

We will be working on the algorithm presented by the paper: A remote-sensing image enhancement algorithm based on patch-wise dark channel prior and histogram equalization with color correction.

The implementation presented by the paper is effective in removing haze from images and is a very well thought out algorithm. It considers all the aspects like sky light, patch size, and scene radiance. It also works on color correction.

In the paper the algorithm was implemented using MATLAB but the problem with using this is it's not available to the normal public. It is not a real time application. Real-time dehazing is used in fast, robust visibility for computer vision applications such as AI robots, auto-pneumatic vehicles, and RS technology. We will implement the algorithm using JAVA which can then be used in an Android app that can be easily accessible to the public and anyone can dehaze images any time.

3. Overview of the Proposed System

3.1. Introduction and Related Concepts

Image dehazing is a poorly caused two-dimensional ill-posed problem in signal or image reconstruction. Its purpose is to restore an unknown image of a hazy image. The model can be expressed as follows:

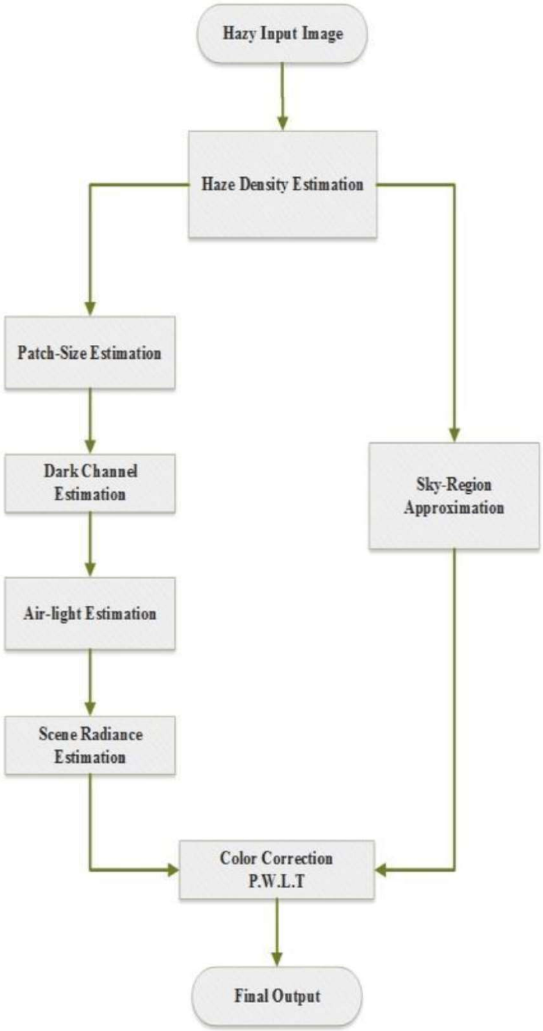
$$I(x)=J(x)t(x)+A(1-t(x)),$$

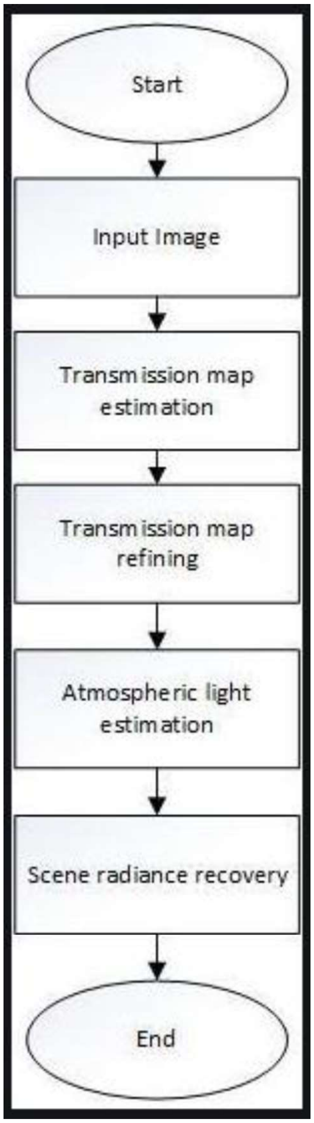
where $I\ (x)$ is the observation image, $J\ (x)$ is the clearer image we need to estimate, A is the global atmospheric light, and $t(x)$ is the medium transmission:

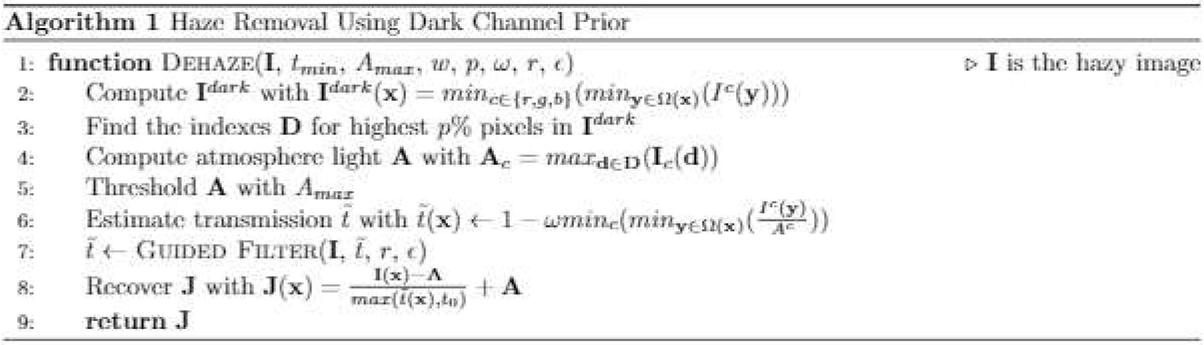
$$J\ (x)=\frac{I(x)-A}{\max(t(x),t_0)}+A.$$

3.2. Framework, Architecture or Module for the Proposed System (with explanation)

The overall flow of our method is shown as

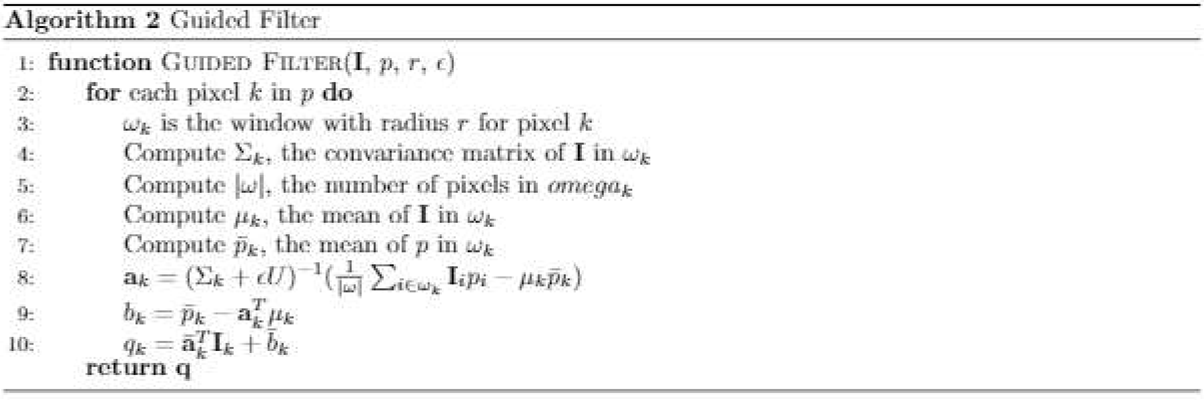



Start
Input Image
Transmission map estimation
Transmission map refining
Atmospheric light estimation
Scene radiance recovery
End



Algorithm 1 Haze Removal Using Dark Channel Prior

```
function DEHAZE($\mathbf{I}$, $t_{min}$, $A_{max}$, $w$, $p$, $\omega$, $r$, $\epsilon$)        ▷ I is the hazy image
    Compute $\mathbf{I}^{dark}$ with $\mathbf{I}^{dark}(\mathbf{x}) = min_{c \in \{r,g,b\}}(min_{\mathbf{y} \in \Omega(\mathbf{x})}(I^c(\mathbf{y})))$
    Find the indexes $\mathbf{D}$ for highest $p\%$ pixels in $\mathbf{I}^{dark}$
    Compute atmosphere light $\mathbf{A}$ with $\mathbf{A}_c = max_{\mathbf{d} \in \mathbf{D}}(\mathbf{I}_c(\mathbf{d}))$
    Threshold $\mathbf{A}$ with $A_{max}$
    Estimate transmission $\tilde{t}$ with $\tilde{t}(\mathbf{x}) \leftarrow 1 - \omega min_c(min_{\mathbf{y} \in \Omega(\mathbf{x})}(\frac{I^c(\mathbf{y})}{A^c}))$
    $\tilde{t} \leftarrow$ GUIDED FILTER($\mathbf{I}$, $\tilde{t}$, $r$, $\epsilon$)
    Recover $\mathbf{J}$ with $\mathbf{J}(\mathbf{x}) = \frac{\mathbf{I}(\mathbf{x}) - \mathbf{A}}{max(\tilde{t}(\mathbf{x}), t_0)} + \mathbf{A}$
    return $\mathbf{J}$
```

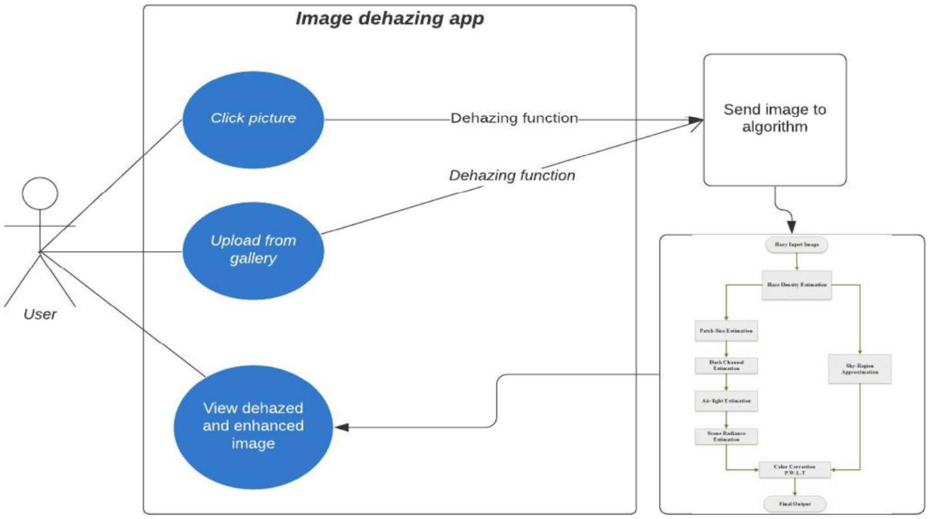


Algorithm 2 Guided Filter

```
function GUIDED FILTER($\mathbf{I}$, $p$, $r$, $\epsilon$)
    for each pixel $k$ in $p$ do
        $\omega_k$ is the window with radius $r$ for pixel $k$
        Compute $\Sigma_k$, the convariance matrix of $\mathbf{I}$ in $\omega_k$
        Compute $|\omega|$, the number of pixels in $omega_k$
        Compute $\mu_k$, the mean of $\mathbf{I}$ in $\omega_k$
        Compute $\bar{p}_k$, the mean of $p$ in $\omega_k$
        $\mathbf{a}_k = (\Sigma_k + \epsilon U)^{-1}(\frac{1}{|\omega|}\sum_{i\in\omega_k}\mathbf{I}_i p_i - \mu_k \bar{p}_k)$
        $b_k = \bar{p}_k - \mathbf{a}_k^T \mu_k$
        $q_k = \bar{\mathbf{a}}_k^T \mathbf{I}_k + \bar{b}_k$
    return q
```

3.3. Proposed System Model (ER Diagram/UML Diagram/Mathematical Modeling)

USE CASE DIAGRAM



4. Proposed System Analysis and Design

4.1. Introduction

Dehazing has a special importance in applications where the images are degraded significantly by the environment. In scenes affected by haze, fog, or smoke, dehazing methods enhance or restore the quality of the images to make them usable in further processing. Even with the emergence of various new dehazing technologies, none of them provide real time support to normal users. We aim to make an app which allows users to dehaze images real time and with high quality and effectiveness.

4.2.Requirement Analysis

Availability:

➢ The system is available all the time.

Efficiency requirements:

➢ **Speed:** The system should be made as fast as possible to reduce response time.

➢ **Throughput:** The throughput should be as high as possible. We should be able to attain maximum output in minimum time.

➢ **Resource Utilization:** Resources are modified according to user requirements

Reliability requirements:

➢ Immediate retrieval of information.

➢ Maintain proper and updated database to improve reliability.

Usability requirements:

➢ Software can be used again and again without distortion.

➢ The interface is simple and easy to use.

➢ System is user friendly.

4.2.1. Functional Requirements

4.2.1.1. Product Perspective: We will be making an android app that takes hazy images as input and returns dehazed good quality images. The app can be used anytime by the user without the need to run a code or any special software.

4.2.1.2. Product features

- ➢ Android app with input at real time image from camera or upload image from gallery
- ➢ Output of android app in dehazed high quality image

4.2.1.3. User characteristics

The different types of users are:

- Surveillance industry workers,
- Transport industry workers: Intelligent vehicles, object recognition etc.
- In consumer photography, the presence of fog will be an annoyance to the images for it reduces the contrast significantly.
- In aerial photography and satellite remote sensing, the photos are much more easily plagued by aerosols.

4.2.1.4. Assumption & Dependencies

1. All these industry workers have an android app with the latest camera feature.
2. They have the app installed and take pictures real time to solve haze issue

4.2.1.5. User Requirements

- Quick response from app without major time delay
- Good quality dehazed image

4.2.2. Non-Functional Requirements

4.2.2.1. <u>Product Requirements</u>

4.2.2.1.1. Efficiency (in terms of Time and Space): Dehazing of images must be real time and the app should give a quick response. The app should be able to process both low size and high size images.

4.2.2.1.2. **Reliability**: The app should give correct dehazed image

4.2.2.1.3. **Portability:** The android app is portable and hence users can use it anytime

4.2.2.1.4. **Usability:** The app should be usable in all hazy conditions. As it is an android app it is usable anytime anywhere.

4.2.2.2. <u>Organizational Requirements</u>

4.2.2.2.1. Implementation Requirements (in terms of deployment)

The app needs to be sent to users in apk format or it needs to be published on Google Play Store so that anyone can download it.

4.2.2.2.2. Engineering Standard Requirements

1. ***Performance Requirements***

The performance requirements for our application are listed below:

- It is important for the app images to be secure
- The application must show smooth transitions between different intents.
- The app must return the dehaze image in real time without too much delay.

2. ***Safety Requirements***

The major safety requirement includes safety of our DB. In case of any catastrophic damage to the DB, the data can get lost. The DB system should recover by using the past backed up data but, still some data might be lost. Thus, it should also run some committed logged transactions to ensure that the data is consistent.

4.2.2.3. Operational Requirements (Explain the applicability for your work

w.r.to the following operational requirement(s))

- **Economic**
- **Environmental**
- **Social**
- **Political**
- **Ethical**
- **Health and Safety**

- **Sustainability**
- **Legality**
- **Inspectability**

4.2.3. System Requirements

4.2.3.1. H/W Requirements (details about Application Specific Hardware)

❑ Android phone with the app installed.

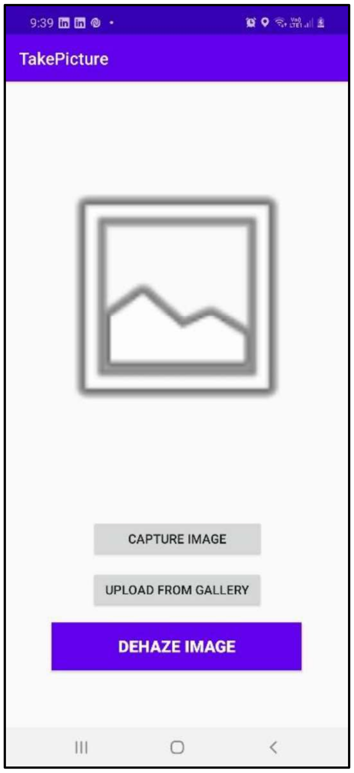
❑ Camera feature in android phone.

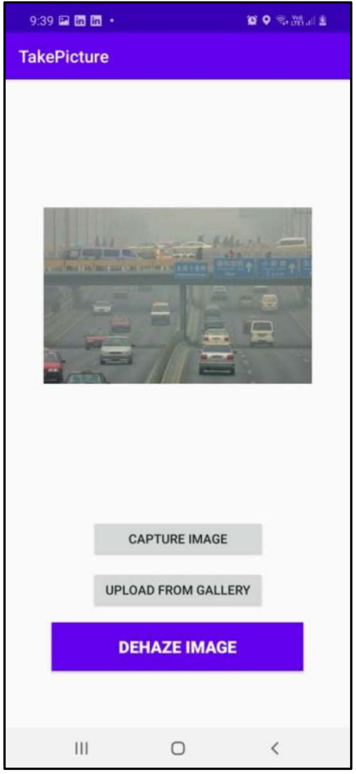
4.2.3.2. S/W Requirements (details about Application Specific Software)

- Python
- OpenCV with java.
- ImShow-Java-OpenCV, a plain image display code implementedby OpenCV Java version.
- Android Studio.
- Google Firebase to save the hazy image and processed image.

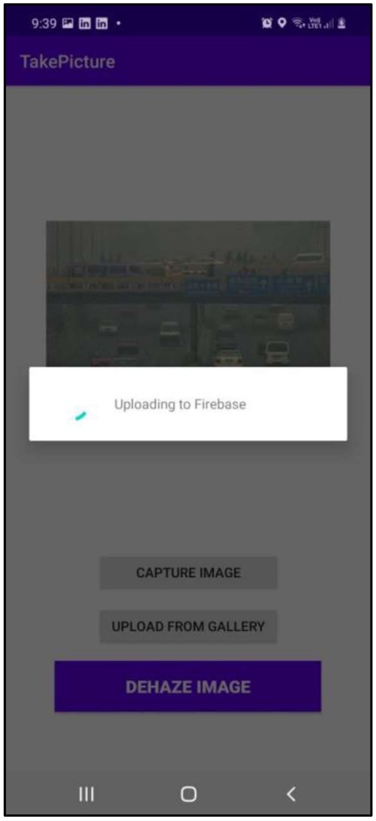
5. Results and discussions

Android app: Our android app has the following features: 1. Taking picture or uploading from gallery.

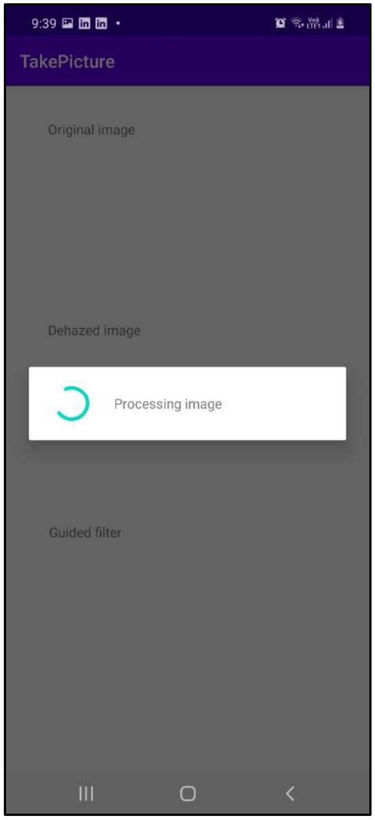




2) Uploading picture to firebase and sending to backend for processing.

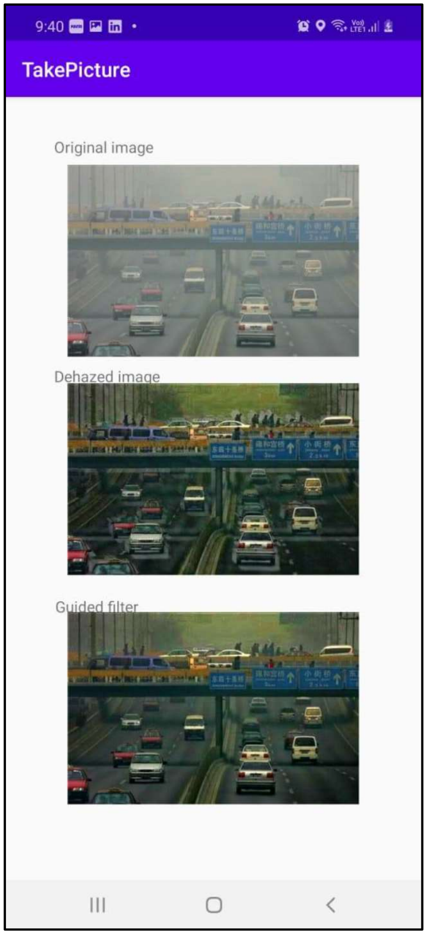


Uploading to Firebase

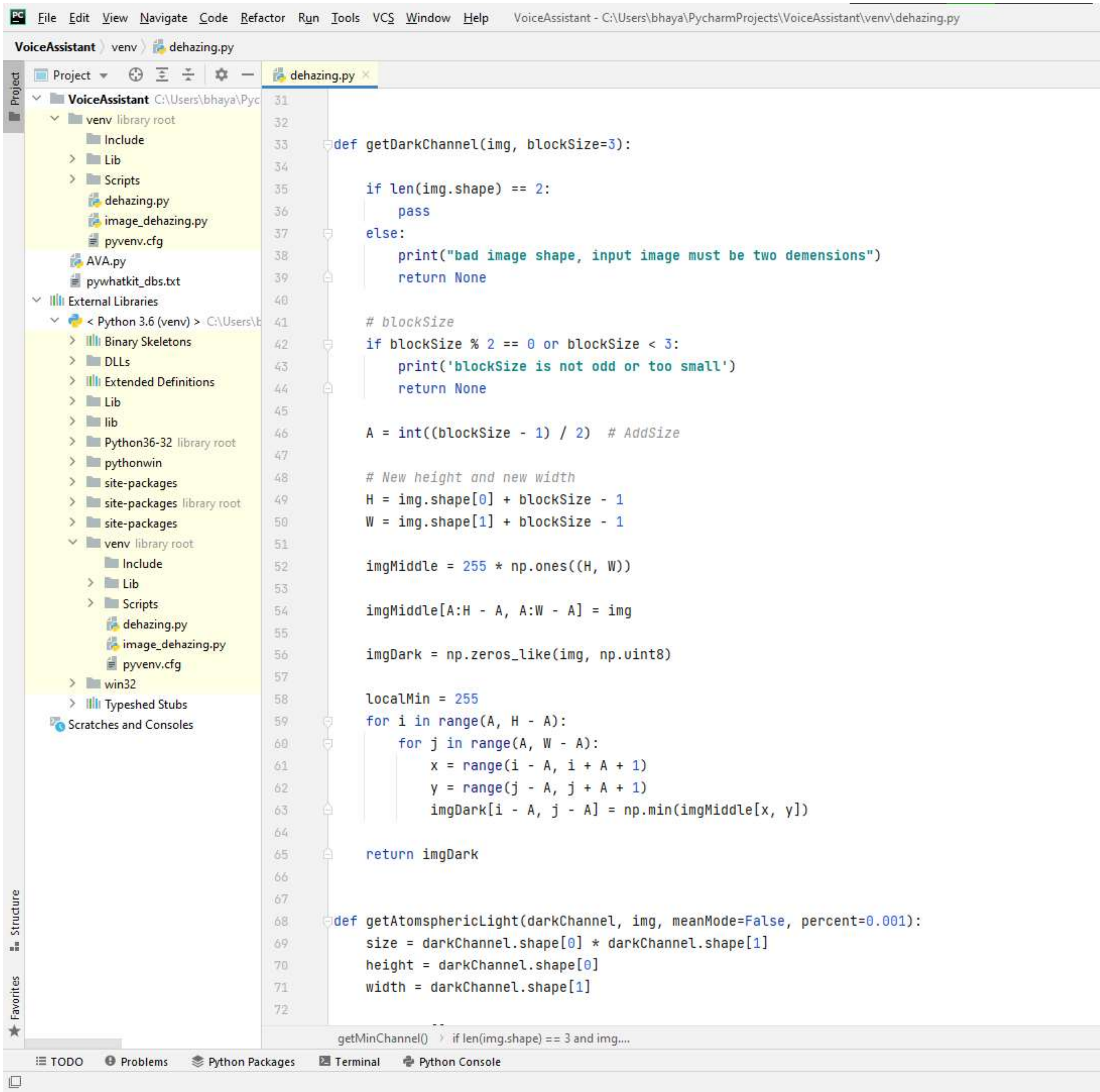


Processing image

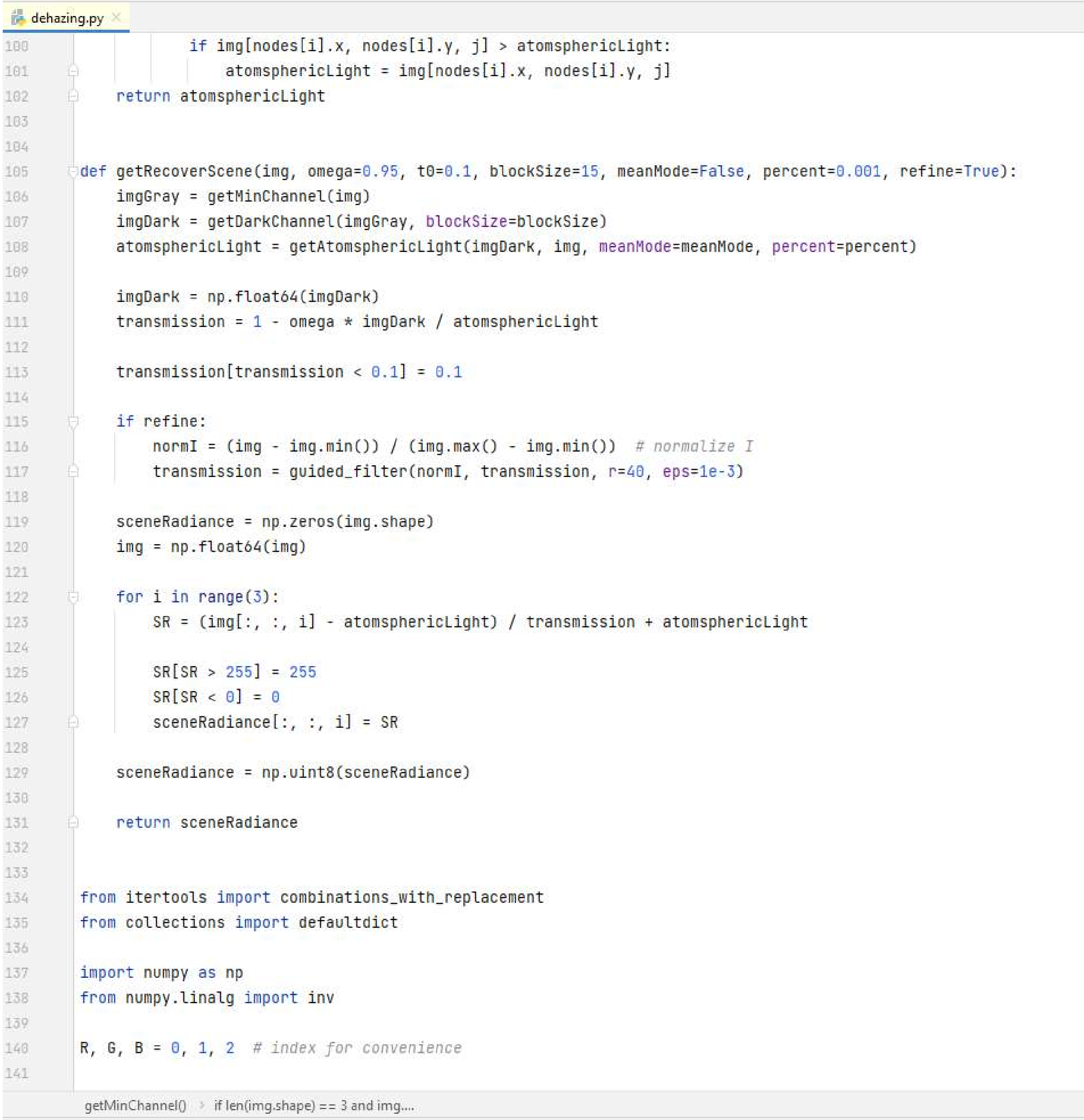
3) Showing hazy image, dehazed image and image after guided filter.



- **Dehazing function:** We are done with 100% of the code for the dehazing algorithm. Different algorithms are used such as *Dark Channel Prior Algorithm, Guided Image Filtering, and Box Filter.* Presented Below are a few instances of the code written in Python, and the Python file is then embedded in the Android App.



```
def getDarkChannel(img, blockSize=3):

    if len(img.shape) == 2:
        pass
    else:
        print("bad image shape, input image must be two demensions")
        return None

    # blockSize
    if blockSize % 2 == 0 or blockSize < 3:
        print('blockSize is not odd or too small')
        return None

    A = int((blockSize - 1) / 2)  # AddSize

    # New height and new width
    H = img.shape[0] + blockSize - 1
    W = img.shape[1] + blockSize - 1

    imgMiddle = 255 * np.ones((H, W))

    imgMiddle[A:H - A, A:W - A] = img

    imgDark = np.zeros_like(img, np.uint8)

    localMin = 255
    for i in range(A, H - A):
        for j in range(A, W - A):
            x = range(i - A, i + A + 1)
            y = range(j - A, j + A + 1)
            imgDark[i - A, j - A] = np.min(imgMiddle[x, y])

    return imgDark


def getAtomsphericLight(darkChannel, img, meanMode=False, percent=0.001):
    size = darkChannel.shape[0] * darkChannel.shape[1]
    height = darkChannel.shape[0]
    width = darkChannel.shape[1]
```



```python
            if img[nodes[i].x, nodes[i].y, j] > atomsphericLight:
                atomsphericLight = img[nodes[i].x, nodes[i].y, j]
    return atomsphericLight


def getRecoverScene(img, omega=0.95, t0=0.1, blockSize=15, meanMode=False, percent=0.001, refine=True):
    imgGray = getMinChannel(img)
    imgDark = getDarkChannel(imgGray, blockSize=blockSize)
    atomsphericLight = getAtomsphericLight(imgDark, img, meanMode=meanMode, percent=percent)

    imgDark = np.float64(imgDark)
    transmission = 1 - omega * imgDark / atomsphericLight

    transmission[transmission < 0.1] = 0.1

    if refine:
        normI = (img - img.min()) / (img.max() - img.min())  # normalize I
        transmission = guided_filter(normI, transmission, r=40, eps=1e-3)

    sceneRadiance = np.zeros(img.shape)
    img = np.float64(img)

    for i in range(3):
        SR = (img[:, :, i] - atomsphericLight) / transmission + atomsphericLight

        SR[SR > 255] = 255
        SR[SR < 0] = 0
        sceneRadiance[:, :, i] = SR

    sceneRadiance = np.uint8(sceneRadiance)

    return sceneRadiance


from itertools import combinations_with_replacement
from collections import defaultdict

import numpy as np
from numpy.linalg import inv

R, G, B = 0, 1, 2  # index for convenience

```

6. References

- **https://ietresearch.onlinelibrary.wiley.com/doi/full/10.1049/ipr2.12004**
- **https://link.springer.com/article/10.1007/s11633-014-0768-7**
- **https://opencv-java-tutorials.readthedocs.io/en/latest/**

Journals

1. Hanspal, R.K. and Sahoo, K., 2017. A survey of image enhancement techniques. *International Journal of Science and Research (IJSR)*, *6*(5), pp.2467-2471.
2. Patel, P. and Bhandari, A., 2019. A review on image contrast enhancement techniques. *Int. J. Online Sci*, *5*(5), pp.14-18.
3. Gupta, P., Kumare, J.S., Singh, U.P. and Singh, R.K., 2017. Histogram based image enhancement techniques: a survey. *Int J Comput Sci Eng*, *5*(6), pp.475-484.
4. Ablin, R., Sulochana, C.H. and Prabin, G., 2020. An investigation in satellite images based on image enhancement techniques. *European Journal of Remote Sensing*, *53*(sup2), pp.86-94.
5. Prasad, K.K. and Aithal, P.S., 2017. A Conceptual Study on Image Enhancement Techniques for Fingerprint Images. *International Journal of Applied Engineering and Management Letters (IJAEML)*, *1*(1), pp.63-72.
6. Ott, A., Mazaheri, A., Lobo, N.D. and Shah, M., 2020, October. Deep Photo Cropper And Enhancer. In *2020 IEEE International Conference on Image Processing (ICIP)* (pp. 993-997). IEEE.
7. Bhandari, Aashish & Kafle, Aayush & Dhakal, Pranjal & Joshi, Prateek Raj & Kshatri, Dinesh. (2020). Image Enhancement and Object Recognition for Night Vision Surveillance.
8. Xu, X., Wang, S., Wang, Z., Zhang, X. and Hu, R., 2020. Exploring Image Enhancement for Salient Object Detection in Low Light Images. *arXiv preprint arXiv:2007.16124*.
9. Kwon, D., Kim, G. and Kwon, J., 2020. DALE: Dark Region-Aware Low-light Image Enhancement. *arXiv preprint arXiv:2008.12493*.
10. Azizi, Z., Lei, X. and Kuo, C.C.J., 2020, December. Noise-Aware Texture-Preserving Low-Light Enhancement. In *2020 IEEE International Conference on Visual Communications and Image Processing (VCIP)* (pp. 443-446). IEEE.

11. Dharejo FA, Zhou Y, Deeba F, Jatoi MA, Du Y, Wang X. A remote-sensing image enhancement algorithm based on patch-wise dark channel prior and histogram equalisation with colour correction. *IET Image Process. 2021;15:47–56.*
12. Sandoub G, Atta R, Ali HA, Abdel-Kader RF. A low-light image enhancement method based on bright channel prior and maximum colour channel. *IET Image Processing. 2021;1–14.*
13. Wonjun Kim, Ryong Lee, Minwoo Park, Sang-Hwan Lee, Myung-Seok Choi. Low-Light Image Enhancement Using Volume-Based Subspace Analysis. *Digital Object Identifier 10.1109/ACCESS.2020.3005249.*
14. Vivek Sharma, Ali Diba, Davy Neven, Michael S. Brown, Luc Van Gool, Rainer Stiefelhagen. Classification-Driven Dynamic Image Enhancement. *Digital Object Identifier 10.1109/ACCESS.2019.*
15. Sean Moran, Pierre Marza, Steven McDonagh, Sarah Parisot, Gregory Slabaugh. DeepLPF: Deep Local Parametric Filters for Image Enhancement. *IEEE Explore, 2020*.