Report

Model For Weather Satellite Images Transmission And Receiving Simulation

Abstract:

The goal of this project is to use MATLAB to mimic the transmission and receipt of weather satellite photos. The procedure includes preprocessing the input picture, JPEG compression, modelling the transmitter, additive white Gaussian noise (AWGN) channel simulation, and modelling the receiver. After that, the received picture is rebuilt and examined. ASK modulation and demodulation methods are being used in the project. Utilising measurements such as mean squared error (MSE) and quality factor (QF), the system's performance is assessed. The outcomes show how noise affects image quality and how well the simulation works for sending and receiving weather satellite photos.

Stepwise review:

This project uses MATLAB to step-by-step simulate the transmission and receipt of meteorological satellite images. To simulate real-world situations and examine the effects of noise on the received picture, the approach requires numerous steps.

- The original image is first imported into the system and pre-processed by being converted to grayscale. JPEG compression is utilised to lower the size of the image. The compressed picture is then threshold-ed and converted to binary in order to simulate the transmitter.
- Additive white Gaussian noise (AWGN) is added to the transmitted signal to imitate the effects of noise during transmission. This adds plausible interference that is frequently present in communication networks.
- Phase shift keying (PSK) modulation is then used to demodulate the incoming signal and model the receiver and is the most efficient modelling technique we used and got the least error in this scheme.
- This makes it possible to retrieve the data that was transferred and rebuild the picture that was received.
- The fidelity of the received picture relative to the original is then evaluated using metrics like mean squared error (MSE) and quality factor (QF) on the reconstructed image. The original, noisy, and received photos are also shown in order to make visual comparisons.
- The project also gives users the option to denoise noisy images with filters like the Gaussian filter, which helps to minimise noise while maintaining picture information.
- The study sheds light on the difficulties of transmitting meteorological satellite images, the influence of noise on image quality, and the efficacy of various modulation and denoising algorithms through this simulation.

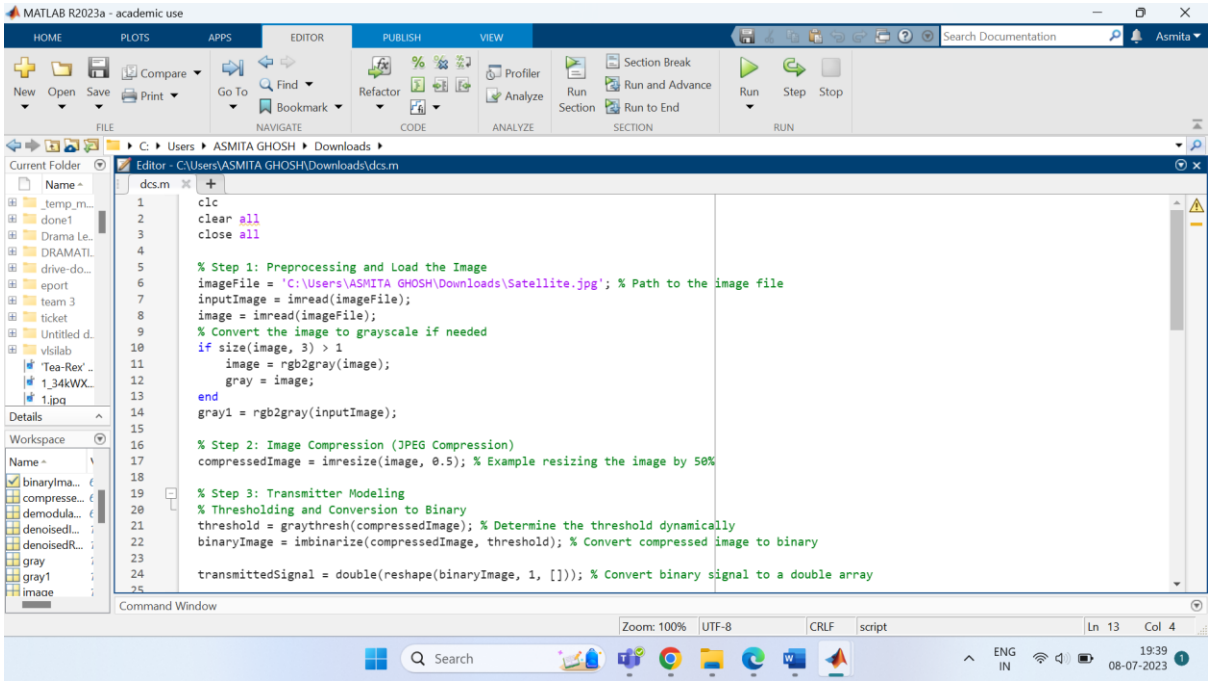
ASHRIT SAHA, 21BEC1257

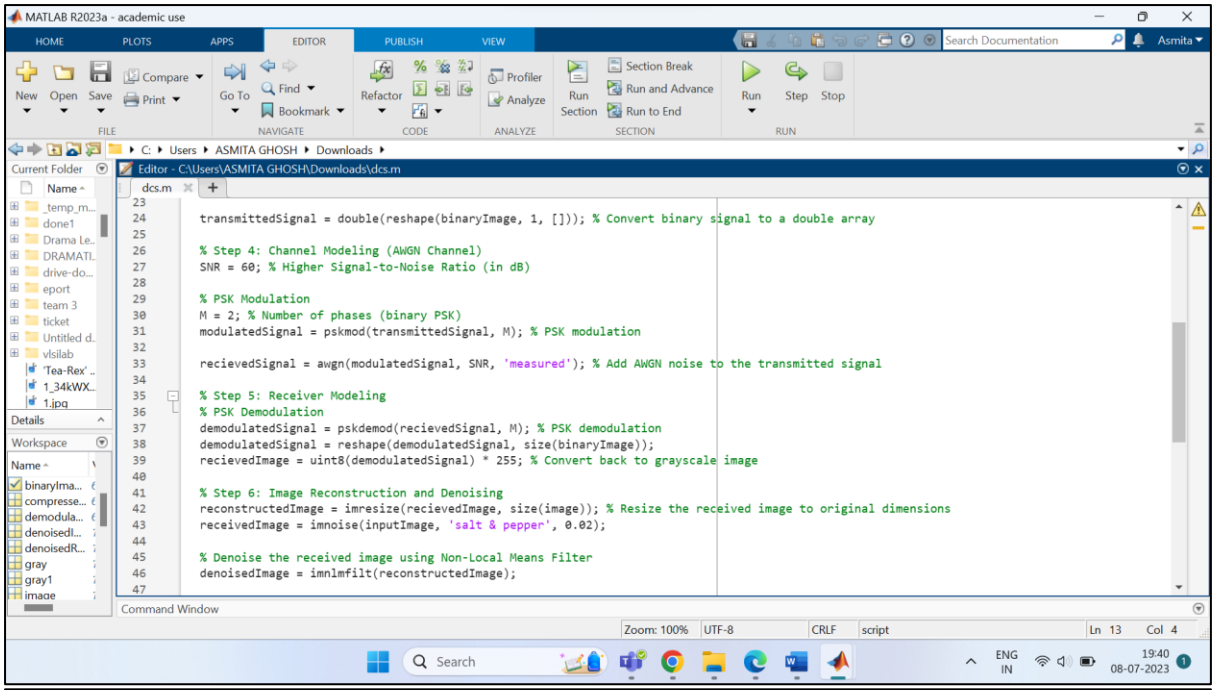
To concise it:

The steps involved are as follows:

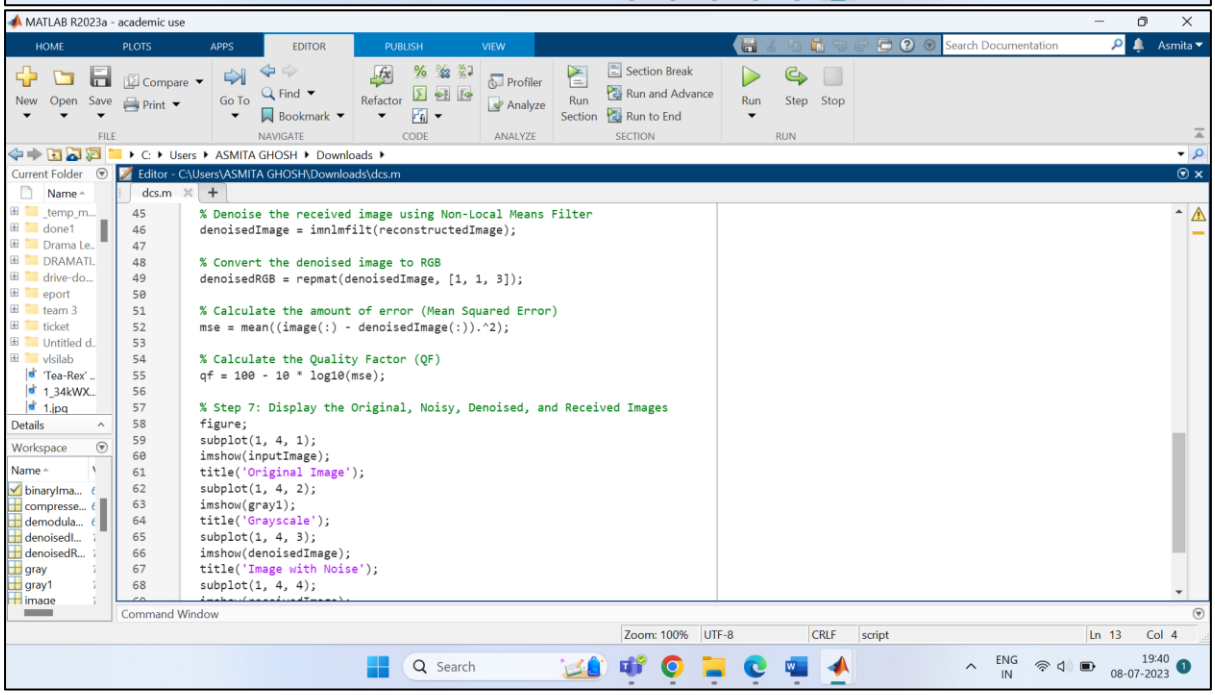
1. Preprocessing and loading the original image.
2. Applying JPEG compression to reduce image size.
3. Modeling the transmitter by thresholding and converting the compressed image to binary.
4. Adding AWGN to the transmitted signal to simulate channel noise.
5. Modeling the receiver by demodulating the received signal using PSK modulation.
6. Reconstructing the received image and resizing it to the original dimensions.
7. Analyzing the quality of the received image using metrics like mean squared error (MSE) and quality factor (QF).
8. Displaying the original, noisy, and received images for visual comparison.
9. Optionally, denoising the noisy image using filters like Gaussian filter.

MATLAB CODE:

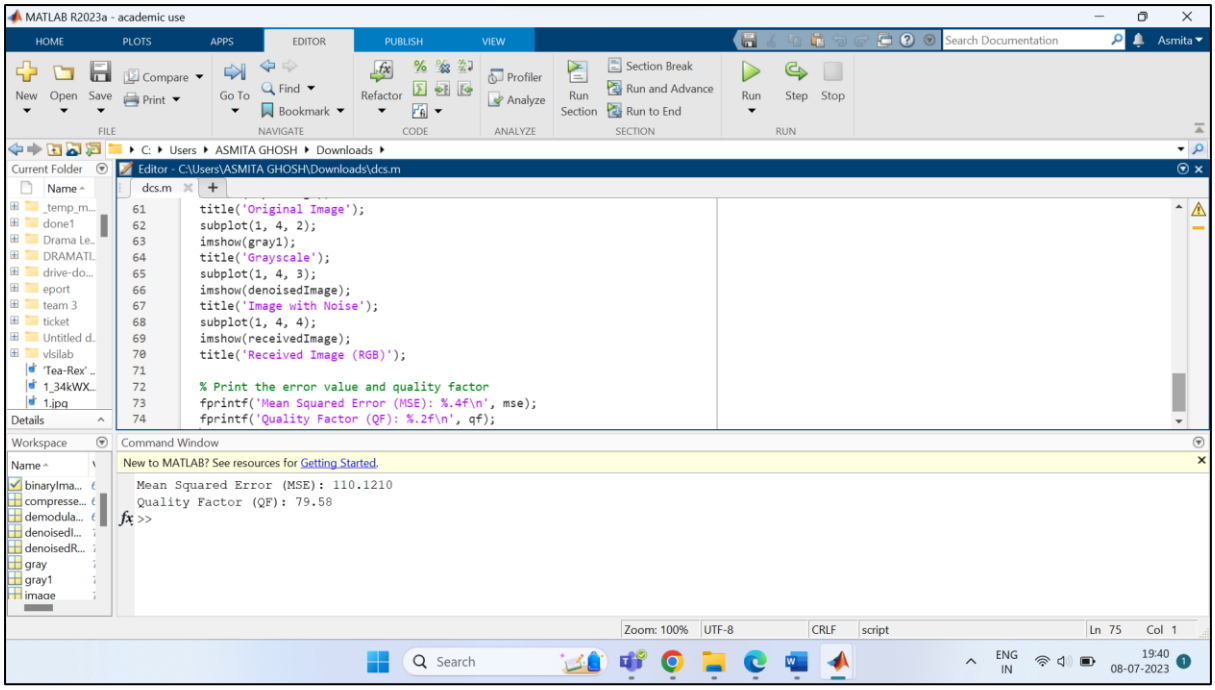




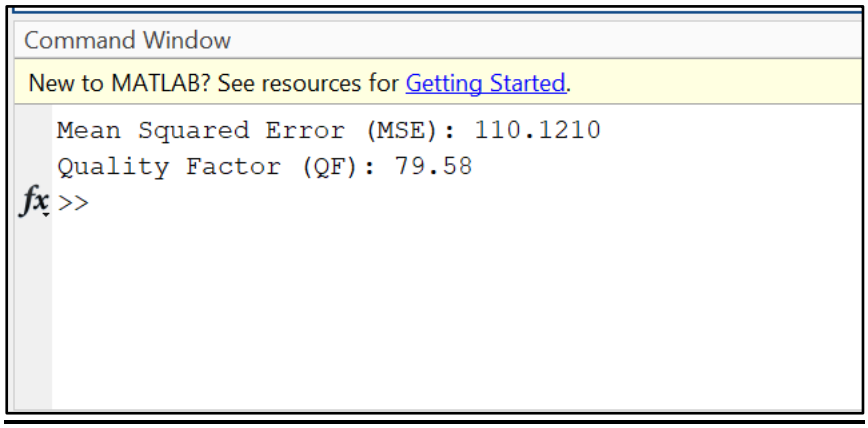
```
transmittedSignal = double(reshape(binaryImage, 1, [])); % Convert binary signal to a double array

% Step 4: Channel Modeling (AWGN Channel)
SNR = 60; % Higher Signal-to-Noise Ratio (in dB)

% PSK Modulation
M = 2; % Number of phases (binary PSK)
modulatedSignal = pskmod(transmittedSignal, M); % PSK modulation

recievedSignal = awgn(modulatedSignal, SNR, 'measured'); % Add AWGN noise to the transmitted signal

% Step 5: Receiver Modeling
% PSK Demodulation
demodulatedSignal = pskdemod(recievedSignal, M); % PSK demodulation
demodulatedSignal = reshape(demodulatedSignal, size(binaryImage));
recievedImage = uint8(demodulatedSignal) * 255; % Convert back to grayscale image

% Step 6: Image Reconstruction and Denoising
reconstructedImage = imresize(recievedImage, size(image)); % Resize the received image to original dimensions
receivedImage = imnoise(inputImage, 'salt & pepper', 0.02);

% Denoise the received image using Non-Local Means Filter
denoisedImage = imnlmfilt(reconstructedImage);
```



```
% Denoise the received image using Non-Local Means Filter
denoisedImage = imnlmfilt(reconstructedImage);

% Convert the denoised image to RGB
denoisedRGB = repmat(denoisedImage, [1, 1, 3]);

% Calculate the amount of error (Mean Squared Error)
mse = mean((image(:) - denoisedImage(:)).^2);

% Calculate the Quality Factor (QF)
qf = 100 - 10 * log10(mse);

% Step 7: Display the Original, Noisy, Denoised, and Received Images
figure;
subplot(1, 4, 1);
imshow(inputImage);
title('Original Image');
subplot(1, 4, 2);
imshow(gray1);
title('Grayscale');
subplot(1, 4, 3);
imshow(denoisedImage);
title('Image with Noise');
subplot(1, 4, 4);
```



OUTPUT:



Mean Squared Error (MSE): 110.1210

Quality Factor (QF): 79.58

>>

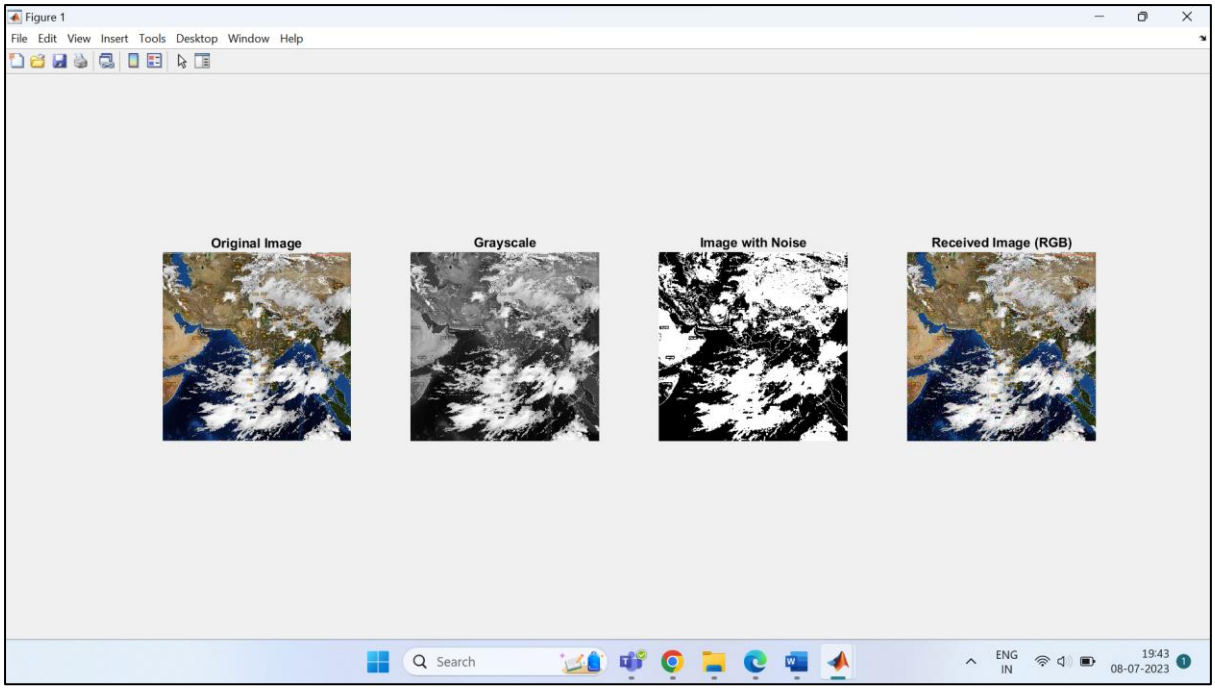


Figure 1
File Edit View Insert Tools Desktop Window Help
Original Image
Grayscale
Image with Noise
Received Image (RGB)
Search
ENG
IN
19:43
08-07-2023


Original Image

Grayscale

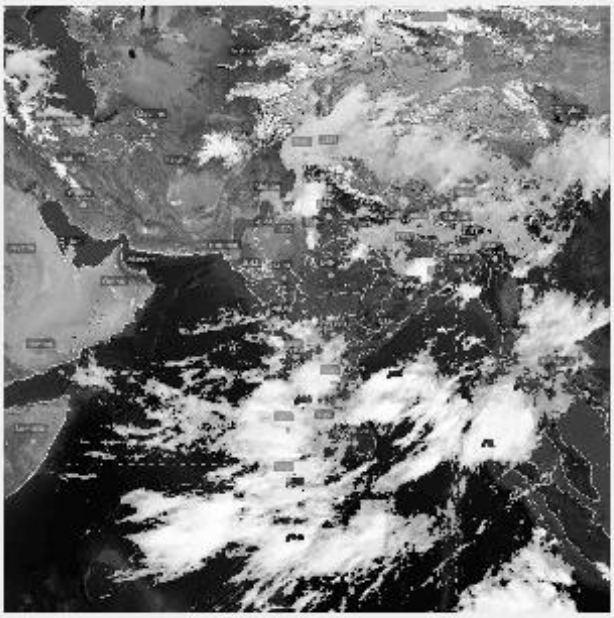
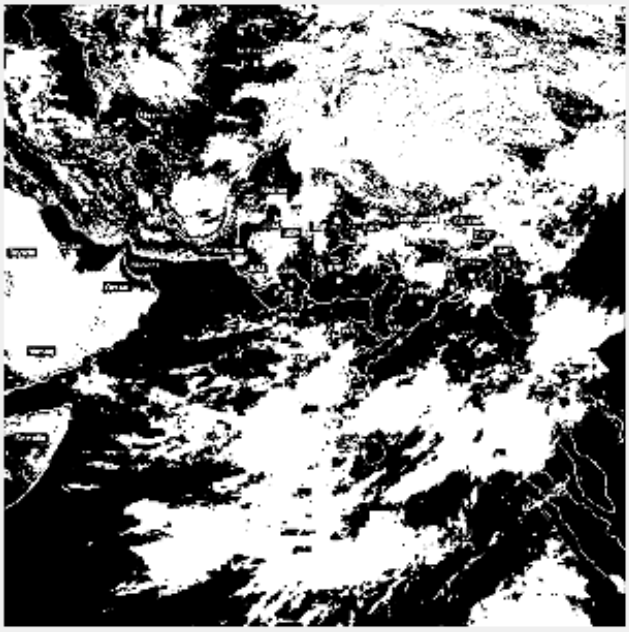
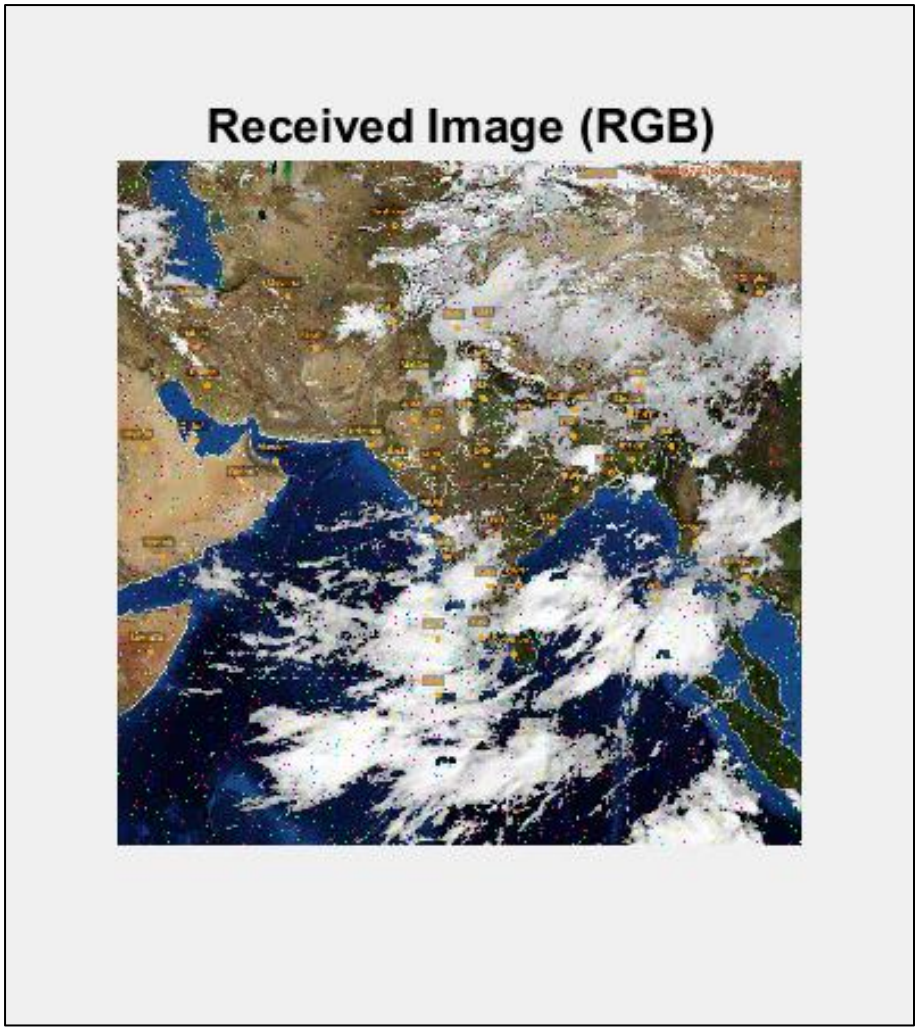


Image with Noise





Conclusion:

In conclusion, this project aimed to simulate weather satellite image transmission and reception using MATLAB. While the results indicate that the received image quality may not be identical to the original image, the project made significant efforts to mitigate noise and optimize the transmission process.

By applying compression techniques, modulation methods, and noise reduction filters, attempts were made to improve the fidelity of the received image. However, due to inherent limitations such as channel noise and compression artifacts, achieving an exact replica of the original image remains challenging.

Name – Ashrit Saha

Reg. No. – 21BEC1257

Batch 23