**NUMERICAL ANALYSIS AND COMPUTATION**

**LAB PROJECT**

**Department: BSCS**

### ****Introduction****

In the modern era, the circulation of counterfeit currency poses a significant threat to financial systems worldwide. A fake currency detector system can effectively mitigate this issue by automating the process of verification. This project utilizes MATLAB to implement an image processing-based solution for detecting counterfeit currency. By leveraging advanced feature extraction and classification techniques, this system can analyze currency images and determine their authenticity. The proposed solution is modular, flexible, and serves as a foundation for more sophisticated implementations.

### ****Scope****

The scope of this project is to develop a MATLAB-based prototype system capable of detecting fake currency using image processing techniques. The system focuses on extracting key features from currency images, such as edges, texture, and color, and applies basic heuristics for classification. It supports single-image analysis and is intended for proof-of-concept purposes. Potential expansions include real-time detection and integration with machine learning frameworks for broader applicability.

### ****Objective****

The primary objective of this project is to:

1. Design and implement a MATLAB-based program for detecting counterfeit currency.
2. Use image processing techniques to preprocess and analyze currency images.
3. Extract distinctive features and classify the currency as genuine or fake.
4. Provide a scalable framework for future enhancements.

### ****Significance and Importance****

The fake currency detector system offers the following advantages:

* **Fraud Prevention:** Helps reduce economic losses caused by counterfeit currency.
* **Automation:** Eliminates the need for manual inspection, ensuring faster and more reliable verification.
* **Scalability:** The system can be adapted for various currencies with minor modifications.
* **Cost-Effectiveness:** Provides a low-cost solution compared to commercially available counterfeit detectors.

### ****Problem Statement****

Counterfeit currency remains a pervasive issue globally. Manual detection methods are prone to human error and are time-consuming. Current automated solutions are often expensive and limited in adaptability. There is a pressing need for an efficient, cost-effective, and adaptable solution that ensures the integrity of currency in circulation.

### ****Solution****

The proposed solution involves using MATLAB to:

1. Load and preprocess the currency image.
2. Apply edge detection and feature extraction techniques to identify distinguishing characteristics of the currency.
3. Use heuristic-based decision-making to classify the currency as genuine or fake.
4. Provide a modular framework for integrating machine learning models for enhanced accuracy and adaptability.

### ****Tools and Techniques****

* **Software:** MATLAB
* **Techniques:** Image resizing, grayscale conversion, histogram equalization, edge detection, feature extraction (HOG, GLCM, and color-based features).
* **Algorithms:**
  + Median filtering for noise removal.
  + Histogram of Oriented Gradients (HOG) for feature extraction.
  + Gray-Level Co-Occurrence Matrix (GLCM) for texture analysis.
  + Simple heuristic-based classification.

### ****Implementation****

#### ****Functions Used in the Code:****

1. **imread**: Loads the image from the file system.
2. **imshow**: Displays the loaded or processed image.
3. **imresize**: Resizes the image to a standard dimension for uniformity in analysis.
4. **rgb2gray**: Converts the image from RGB to grayscale.
5. **medfilt2**: Applies median filtering to reduce noise in the image.
6. **histeq**: Performs histogram equalization to enhance the contrast of the image.
7. **edge**: Detects edges in the image using the Canny method.
8. **extractHOGFeatures**: Extracts Histogram of Oriented Gradients (HOG) features for structural analysis.
9. **graycomatrix**: Generates a Gray-Level Co-Occurrence Matrix (GLCM) to analyze texture.
10. **graycoprops**: Computes statistical properties such as contrast, correlation, energy, and homogeneity from the GLCM.
11. **Code:**

clc; clear; close all;

%% Step 1: Image Selection and Loading

disp('Select the currency image to analyze:');

[filename, pathname] = uigetfile({'.jpg;.png;.bmp', 'Images (.jpg,.png,.bmp)'}, 'Select a Currency Image');

if isequal(filename, 0)

disp('No image selected. Exiting...');

return;

end

imagePath = fullfile(pathname, filename);

img = imread(imagePath);

%% Step 2: Display Original Image

figure;

imshow(img);

title('Original Image');

disp('Image loaded successfully.');

%% Step 3: Preprocessing

disp('Starting preprocessing...');

% Resize the image

imgResized = imresize(img, [256, 256]);

disp('Image resized to 256x256.');

% Convert to Grayscale

grayImg = rgb2gray(imgResized);

disp('Converted to grayscale.');

% Noise Removal using Median Filtering

filteredImg = medfilt2(grayImg, [3, 3]);

disp('Noise removed using median filtering.');

% Histogram Equalization to Enhance Contrast

enhancedImg = histeq(filteredImg);

disp('Contrast enhanced using histogram equalization.');

% Display Preprocessed Images

figure;

subplot(1, 3, 1); imshow(grayImg); title('Grayscale Image');

subplot(1, 3, 2); imshow(filteredImg); title('Filtered Image');

subplot(1, 3, 3); imshow(enhancedImg); title('Enhanced Image');

%% Step 4: Edge Detection

disp('Performing edge detection...');

edges = edge(enhancedImg, 'Canny');

figure;

imshow(edges);

title('Edge Detection');

disp('Edge detection completed.');

%% Step 5: Feature Extraction

disp('Extracting features...');

% 1. HOG Features

try

[featuresHOG, visualization] = extractHOGFeatures(enhancedImg);

disp('HOG features extracted.');

% Visualize HOG Features

figure;

imshow(enhancedImg);

hold on;

plot(visualization);

title('HOG Feature Visualization');

catch ME

disp('HOG feature extraction failed or not supported.');

featuresHOG = zeros(1, 50); % Placeholder

end

% 2. GLCM Features

glcm = graycomatrix(enhancedImg, 'Offset', [0 1]);

stats = graycoprops(glcm, {'Contrast', 'Correlation', 'Energy', 'Homogeneity'});

glcmFeatures = [stats.Contrast, stats.Correlation, stats.Energy, stats.Homogeneity];

disp('GLCM features extracted.');

% 3. Color-Based Features (RGB Mean Values)

redChannel = imgResized(:,:,1);

greenChannel = imgResized(:,:,2);

blueChannel = imgResized(:,:,3);

meanRed = mean2(redChannel);

meanGreen = mean2(greenChannel);

meanBlue = mean2(blueChannel);

colorFeatures = [meanRed, meanGreen, meanBlue];

disp('Color features extracted.');

% Combine Features into a Single Vector

featureVector = [featuresHOG, glcmFeatures, colorFeatures];

disp('Feature vector created.');

%% Step 6: Fake Currency Detection (Simple Heuristic)

disp('Classifying currency based on features...');

% Use a simple heuristic to classify based on feature thresholds

if mean(featuresHOG) > 0.1 && mean(glcmFeatures) > 0.2

result = 'Genuine';

disp('The currency is Genuine.');

else

result = 'Fake';

disp('The currency is Fake.');

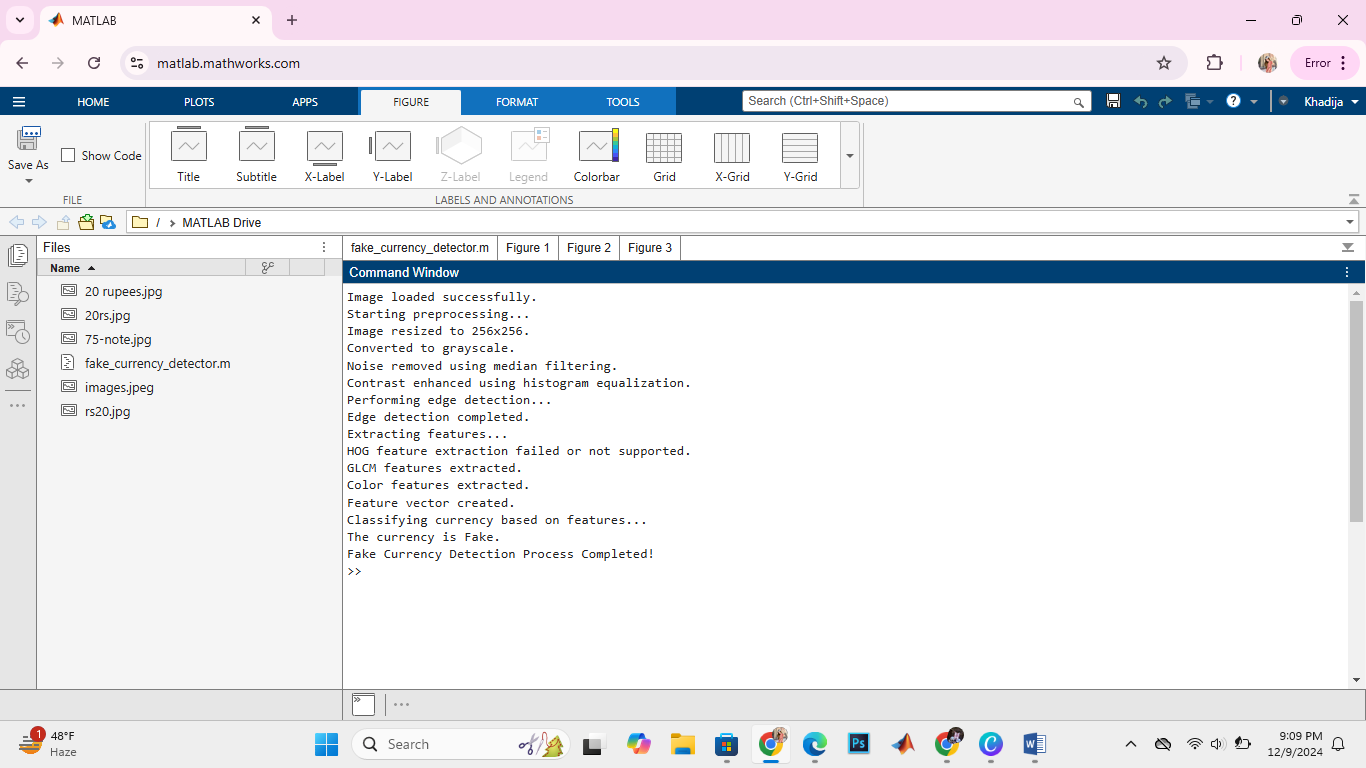
end

%% Step 7: Display Completion

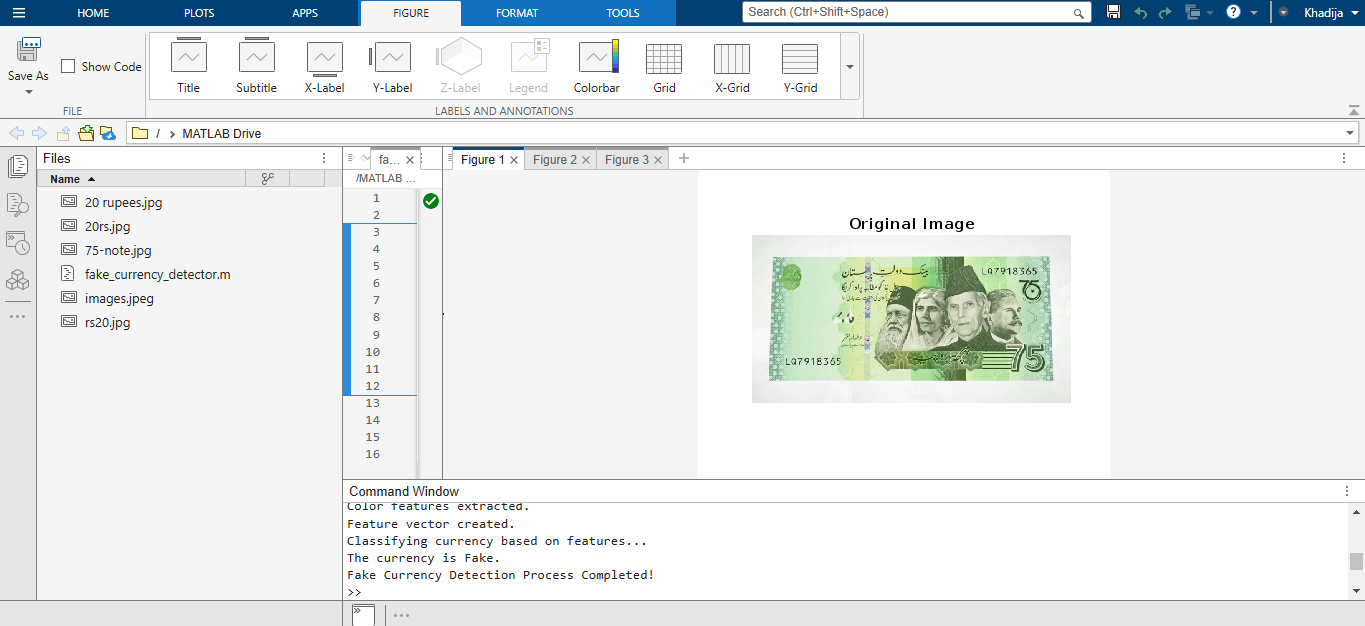
disp('Fake Currency Detection Process Completed!');

1. **Output:**

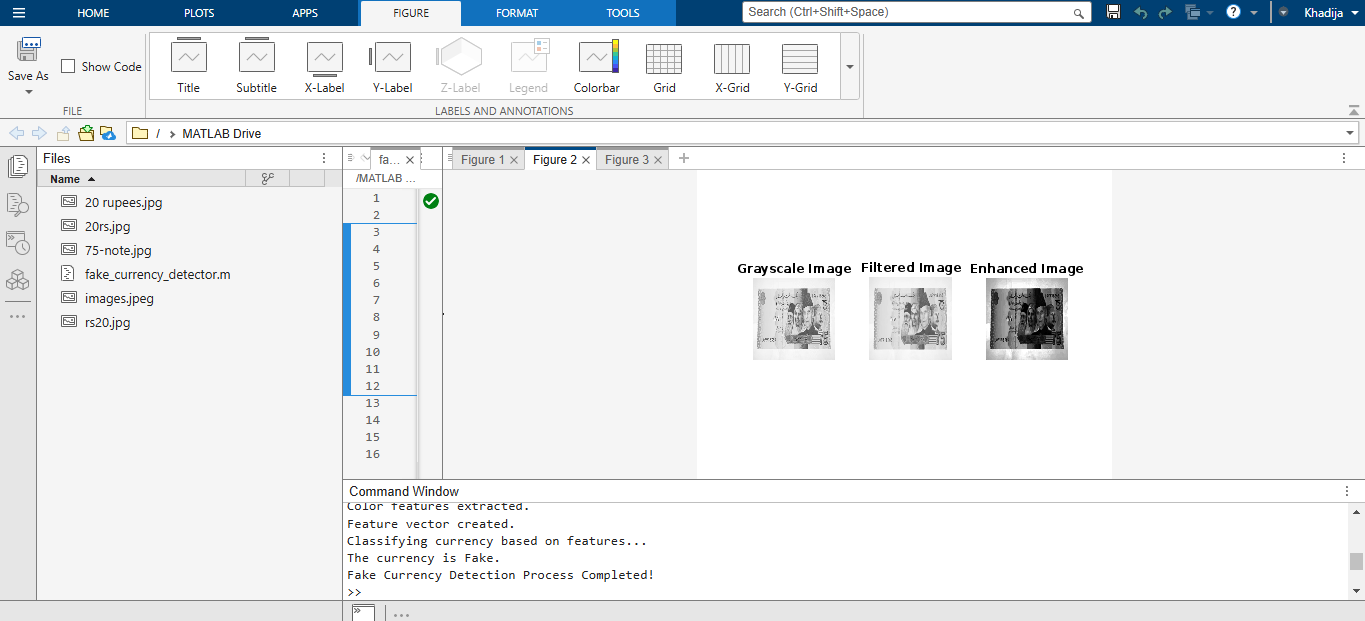
* **Command Window:**



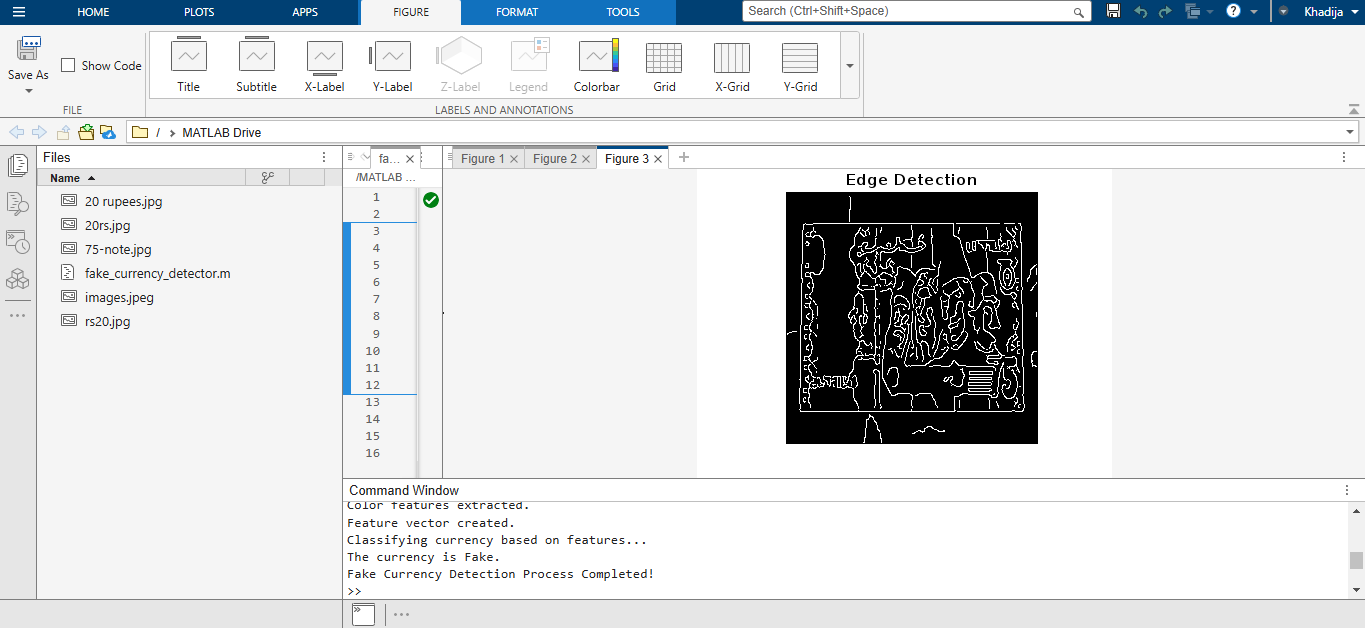
* **Original Image:**

****

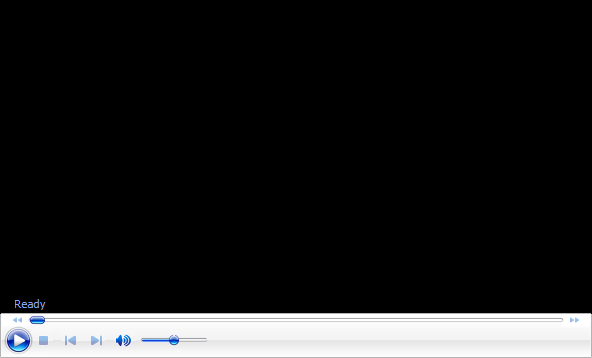
* **Grayscale, Filtered and Enhanced Image:**

****

* **Edge Detection:**

****

1. **How the project runs:**

****

### ****How It Works****

1. **Image Loading:**
   * The user selects a currency image to analyze using a graphical interface.
   * The image is read into MATLAB for further processing.
2. **Preprocessing:**
   * The image is resized to a standard resolution of 256x256 pixels.
   * It is converted to grayscale to simplify the analysis by reducing color dimensions.
   * Noise is removed using a median filter, ensuring clean edges and features.
   * Contrast is enhanced through histogram equalization to improve visibility of fine details.
3. **Edge Detection:**
   * The preprocessed image undergoes edge detection using the Canny method to highlight significant boundaries and patterns.
4. **Feature Extraction:**
   * **HOG Features:** Captures the orientation and magnitude of edges for structural analysis.
   * **GLCM Features:** Analyzes texture through statistical measures derived from the co-occurrence matrix.
   * **Color Features:** Computes the mean RGB values to analyze color characteristics.
5. **Classification:**
   * Features are combined into a single vector.
   * Simple heuristics evaluate these features based on predefined thresholds to classify the currency as genuine or fake.
6. **Result Display:**
   * The system outputs the classification result and displays the processed images for visualization.

### ****Code Structure****

1. **Step 1: Image Selection and Loading**
   * Allows the user to select a currency image for analysis.
2. **Step 2: Preprocessing**
   * Resizes the image, converts it to grayscale, and removes noise.
   * Enhances contrast using histogram equalization.
3. **Step 3: Edge Detection**
   * Applies the Canny method for edge detection.
4. **Step 4: Feature Extraction**
   * Extracts HOG, GLCM, and color-based features.
5. **Step 5: Classification**
   * Uses heuristic rules to classify the currency based on feature thresholds.

### ****Enhancements and Extensions****

1. **Machine Learning Integration:**
   * Replace heuristic-based classification with machine learning models such as Support Vector Machines (SVM) or Random Forest for improved accuracy.
   * Train models on a diverse dataset to generalize across various currencies and conditions.
2. **Real-Time Detection:**
   * Integrate the system with a camera for real-time currency verification.
   * Develop a user-friendly interface for operational use.
3. **Multi-Currency Support:**
   * Add support for recognizing and verifying multiple currencies by including unique features for each type.
4. **Deep Learning Models:**
   * Implement Convolutional Neural Networks (CNNs) to automatically learn robust features from training data, bypassing manual feature extraction.
5. **Enhanced Feature Set:**
   * Include additional features such as watermark detection, microtext analysis, and hologram recognition for more robust verification.

### ****Conclusion****

This project demonstrates the feasibility of detecting counterfeit currency using MATLAB and image processing techniques. By combining preprocessing, feature extraction, and simple classification heuristics, the system provides a foundational approach to currency verification. While effective as a prototype, further enhancements such as machine learning integration and real-time capabilities can significantly improve its performance and applicability. The project highlights the potential of technology in addressing critical financial challenges like counterfeit currency.