User Manual: Facial Feature Detection Tool

Made as a part of the course project for the Computer Vision course at the University of Jordan

Introduction:

Welcome to the user manual for our Facial Feature Detection Tool. This tool is an embodiment of the sophisticated yet accessible world of image processing, designed to provide an intuitive and effective way to detect and analyze facial features in images.

In the realm of facial feature detection, the Viola-Jones algorithm stands as a landmark achievement, particularly noted for its use in machine learning for real-time face detection. While such machine learning algorithms offer powerful capabilities, our tool takes a different approach. Our focus has been to harness and apply fundamental image processing techniques that can be learned and understood without delving into the complexities of machine learning algorithms. This decision is rooted in our commitment to create a tool that aligns more closely with what we have studied and mastered in the course so far, offering a practical application of traditional image processing methods.

System Requirements

- MATLAB (version [specify the required version or higher])
- Image Processing Toolbox
- Computer Vision Toolbox

Getting Started

1. **Launching the Tool:**
    - Open MATLAB.
    - Navigate to the project directory.
    - Run **main** to start the application.
2. **Main Interface:**
    - The interface consists of two image frames (for the original and processed images), several sliders to adjust detection parameters, checkboxes for feature selection, and buttons for image input.

3. **Image Input:**
    - Use the 'Select Image' button to choose an image file from your computer.
    - Or, you can use the 'WebCam' button to capture a Realtime video using your webcam.

Preprocessing Steps in Image Analysis:

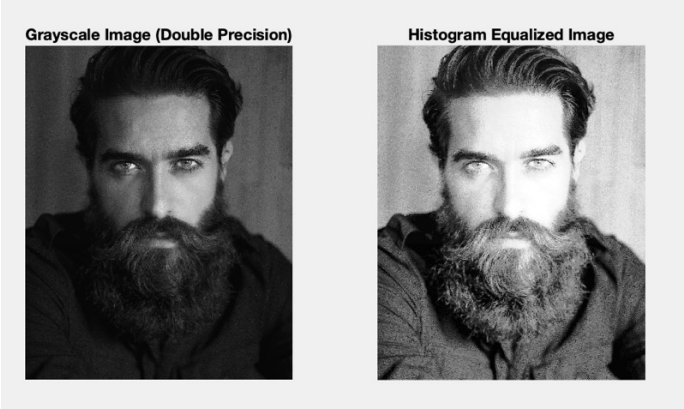
Before detecting facial features, the tool performs several preprocessing steps on the input image. These steps are crucial for enhancing image quality and preparing it for accurate feature detection. Here's a breakdown of each preprocessing step:

1. Conversion to Grayscale and Double Precision

- **Description:** Converts the original RGB image to grayscale, reducing it to a 2D matrix. Then, changes the data type to double precision, which is often required for further image processing tasks.
- **Function: I = im2double(rgb2gray(I));**

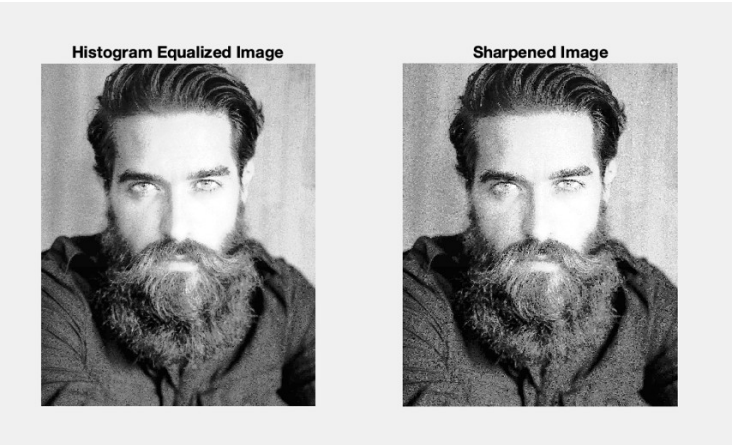
2. Histogram Equalization

- **Description:** Enhances the contrast of the image by spreading out the most frequent intensity values, making features more distinguishable.
- **Function: `I = histeq(I);`**



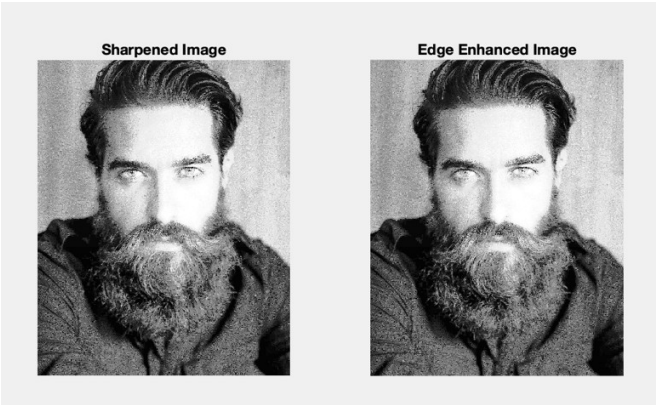
3. Image Sharpening

- **Description:** Enhances edges to make them more distinct, aiding in feature extraction and detection.
- **Function: `I = imsharpen(I);`**

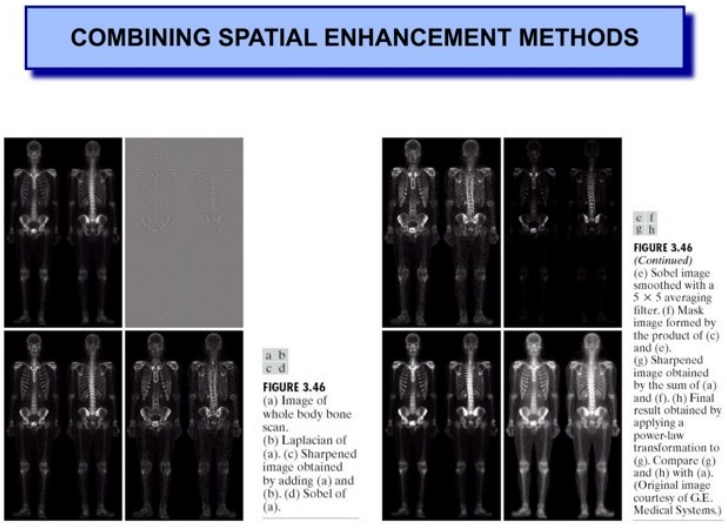


4. Edge Enhancement using Laplacian Filter and Sobel Operator

- **Description:** Applies a Laplacian filter for highlighting edges and details, and uses the Sobel operator to detect edges. This combination emphasizes edges and fine details in the image.
- **Function: I = I + (I + imfilter(I, fspecial('laplacian', 1), 'replicate')) .* imfilter(edge(I, 'sobel'), fspecial('average', [5 5]), 'replicate');**

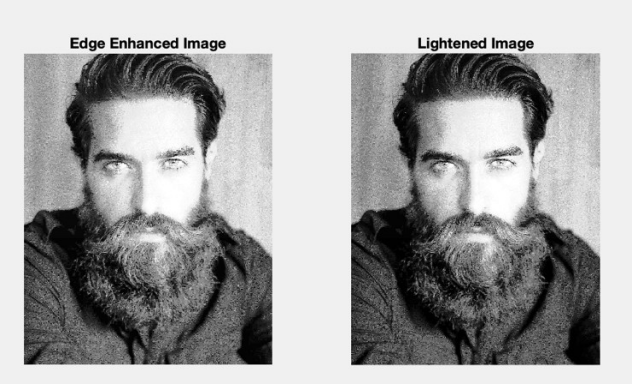


Note that this was inspired by a technique mentioned in the course's slides, find relevant slide below.



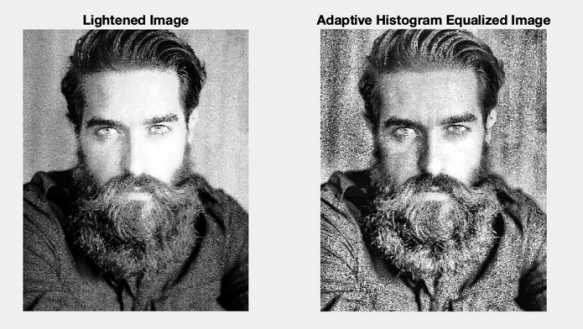
5. Contrast Adjustment (Lightening)

- **Description:** Adjusts the image intensity values to increase brightness, controlled by the **defaultLighten** parameter.
- **Function: I = imadjust(I, [], [], defaultLighten);**



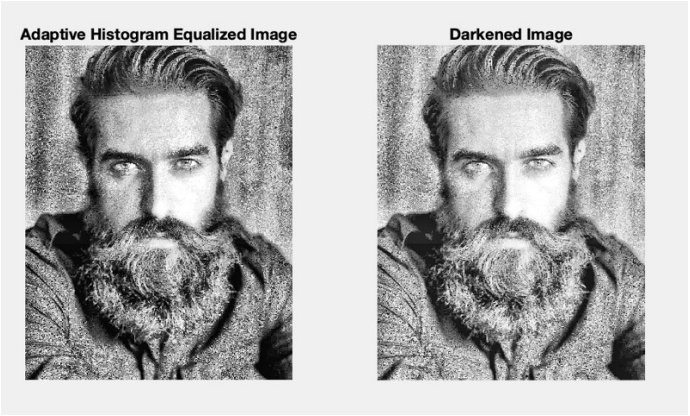
6. Adaptive Histogram Equalization

- **Description:** Enhances local contrast, particularly in regions of lower contrast, which helps in revealing hidden details in darker areas.
- **Function: I = adapthisteq(I);**



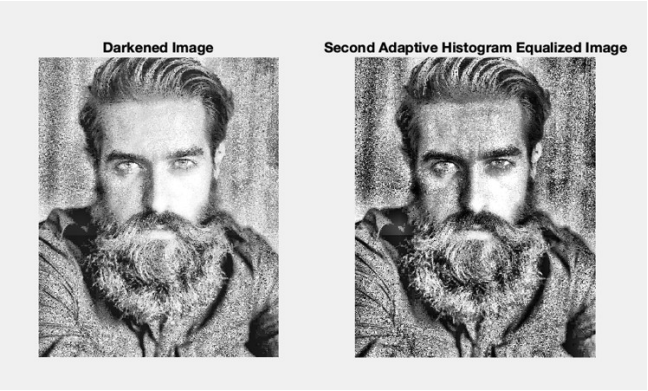
7. Contrast Adjustment (Darkening)

- **Description:** Performs another round of intensity adjustment to potentially darken the image, controlled by the **defaultDarken** parameter.
- **Function: I = imadjust(I, [], [], defaultDarken);**



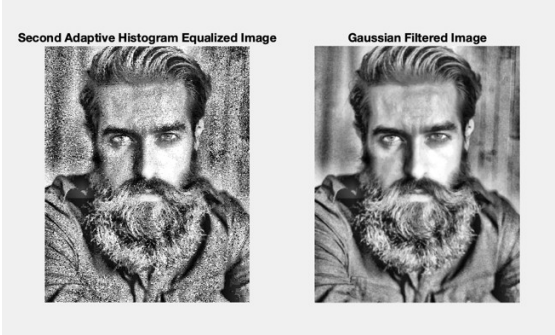
8. Adaptive Histogram Equalization (Again)

- **Description:** A second application of adaptive histogram equalization to further enhance local contrast after the darkening step.
- **Function: I = adapthisteq(I);**



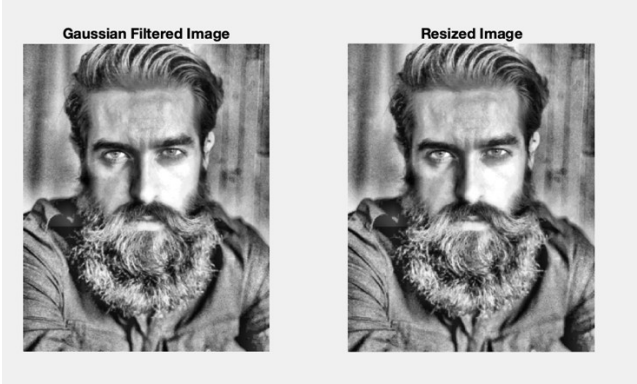
9. Gaussian Filtering

- **Description:** Applies Gaussian smoothing to reduce noise and detail in the image. This is helpful for reducing the effects of noise on edge detection or other analyses. The amount of smoothing is controlled by **gamma**.
- **Function: I = imgaussfilt(I, gamma);**



10. Image Resizing

- **Description:** Changes the size of the image by the factor **scale**. This is useful for standardizing input sizes for further processing or analysis.
- **Function: I = imresize(I, scale);**



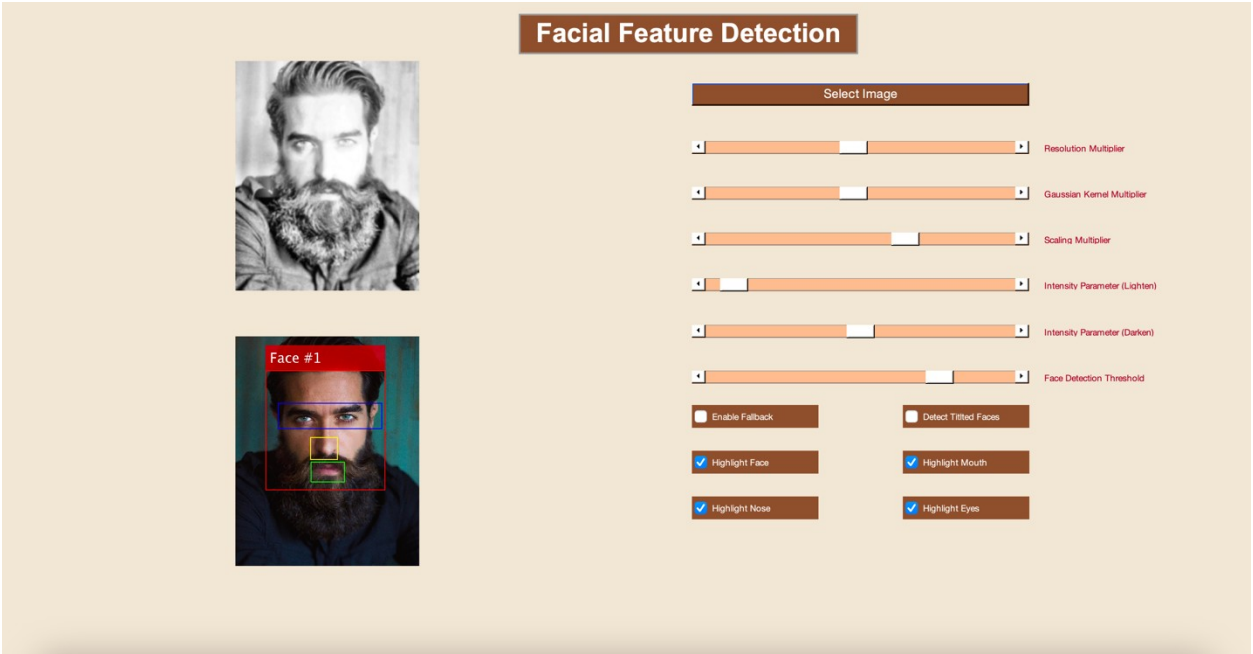
Each of these preprocessing steps prepares the image in a specific way, beneficial for the accurate detection of facial features. The effects of these steps can vary depending on the characteristics of the input image.

Feature Detection:

1. **Adjusting Parameters:**
    - **Resolution Slider:** Adjusts the resolution of the image processing.
    - **Gamma Slider:** Modifies the gamma correction.
    - **Scale Slider:** Changes the scaling factor for the image.
    - **Lighten/Darken Sliders:** Controls the brightness adjustments.
    - **Face Threshold Slider:** Sets the threshold for face detection sensitivity.

2. **Selecting Features:**
    - Use the checkboxes to select which facial features to detect: Face, Eyes, Mouth, and Nose.
    - There are also two extra checkboxes turned off by default, one called fallback, which falls back to geometrical positioning in case Viola-Jones fails to detect the features; the other is for detecting tilted faces at all angles (inclusing upside down faces!).

3. **Viewing Results:**
    - The processed image with detected features will be displayed in the 'newImageFrame' area.
    - Detected features are highlighted with colored rectangles (red for the face, green for the mouth, yellow for the nose, and blue for the eyes).



Troubleshooting

- **No Features Detected:** Adjust the sliders to fine-tune the detection parameters.
- **Incorrect Detection:** Ensure the image is clear and well-lit.

Conclusion

Thank you for choosing our Facial Feature Detection Tool. This tool represents a blend of advanced image processing techniques and user-friendly design, making it an efficient solution for analyzing and extracting facial features from images.We encourage you to explore various settings and parameters to achieve the best results tailored to your specific needs.