Project 1

Due: 10/11/23

Team Member Names:

Instructions:

1. Fill in your name and attach this page as cover sheet for your report.
2. This is a group project. Work with your group as assigned in the "ECE4510 F23 Project Groups" document. List the contribution from each team member to this report, in the following page. Submit a single report for the group.
3. Attach all supporting material with the report:
    a. MATLAB code and output printout.
    b. OpenCV code and output printout from Jupyter Notebook.
    c. Images, diagrams, figures, etc.
4. Type / handwrite and sign the following honor pledge:
   "I pledge on my honor that I have not given or received any unauthorized assistance on this assignment."

Honor Pledge:

Signature

Grading Information:

Task 1: MATLAB and Averaging Filter	(20 Points)	
Task 2: MATLAB and Linear / 2D filters	(20 Points)	
Task 3: OpenCV and Basic Image Functions	(30 Points)	
Task 4: OpenCV and Basic Video Functions	(30 Points)	
	Total	**/ 100 Points**

Team member contributions to the report:

Team member 1 name:

Contribution:

Team member 2 name:

Contribution:

Team member 3 name:

Contribution:

Team member 4 name:

Contribution:

Project 1 - Task 1

MATLAB and Averaging Filter

The purpose of this project is to get familiar with the image processing toolbox in MATLAB. Review the following basic functions in MATLAB before working on this project:

```
imread      % Read image from graphics file
imshow      % Display image
imshowpair  % Display pair of images for comparison
imfilter    % N-D filtering of multidimensional images
```

a) Process the supplied “Lizard.png” grayscale image using a 5 x 5 averaging or box filter with a weight of 1. Use the `imfilter` command to perform the correlation or convolution operation (both have same result for the averaging filter of equal weights).

i) Display the original and filtered images.

ii) Was the resulting image sharp or blurry when compared with the original? Why?

b) Subtract the filtered image you obtained in part (a) from the original image.

i) Display the resulting image.

ii) How does the resulting image look like? Why?

c) Add two times the image you obtained in part (b) back to the original image.

i) Display the resulting image.

ii) Was the resulting image sharp or blurry when compared with the original? Why?

d) Repeat the process in parts (a) through (c) using an 10 x 10 averaging filter of unit weight.

i) Display the resulting images.

ii) How do the results compare with ones obtained in parts (a) through (c)?

Project 1 - Task 2

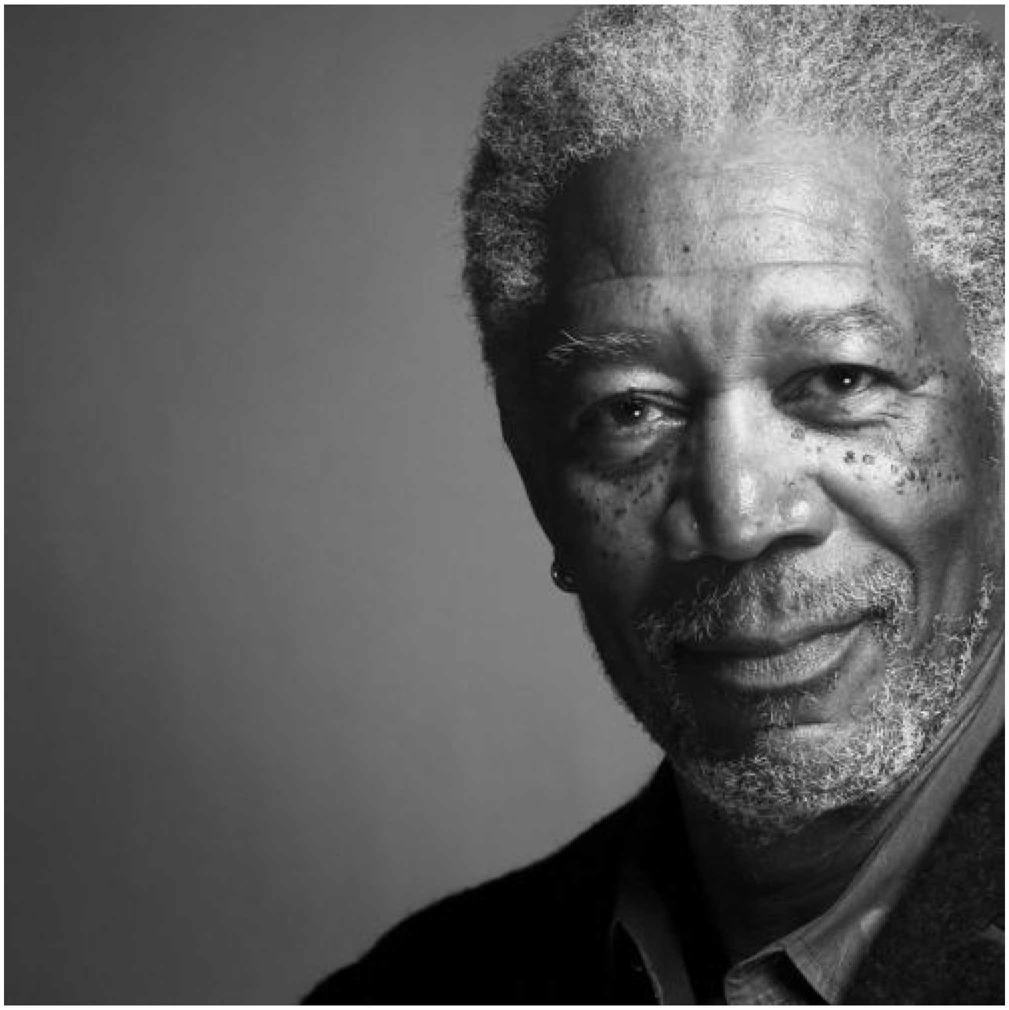
MATLAB and Linear / 2D Filters

The purpose of this project is to get familiar with the image processing toolbox in MATLAB. Review the following basic functions in MATLAB before working on this project:

```
fspecial    % Create predefined 2-D filter
imsharpen   % Sharpen image using unsharp masking
subplot     % Create axes in tiled positions
truesize    % Adjust display size of image
figure      % Create figure window
```

Process the supplied "Morgan.tif" grayscale image by convolving it with the following 3 types of 7 x 7 filters that you create using the `fspecial` function:

1. 'average'	Averaging filter
2. 'disk'	Circular averaging filter
3. 'gaussian'	Gaussian lowpass filter



Use the image sharpening technique used in Task 1 by subtracting the resulting blurred image from the original and adding the result back to original image. Perform this technique using all 3 types of filters separately one at a time. Finally, sharpen the image using unsharp masking with the `imsharpen` function.

a) Display the original and the resulting 3 filtered images in a 2x2 grid using the subplot command with each image magnified to 300x300 pixels using the truesize command.

b) Zoom in on the right eye in the image by creating a 100x100 pixel sub-image from each of the original and filtered images. The top left corner of the sub image is at location (150, 350) and the bottom right corner is at the location (249, 449). Display the resulting 4 images in a 2x2 grid using the subplot command with each image magnified to 300x300 pixels using the truesize command. Compare the filtered images with the original image and comment on the relative effect of each filter.

c) Subtract the filtered image from the original and add the result back to original image to sharpen it. Perform this technique using all 3 filtered images separately one at a time. Finally, sharpen the image using unsharp masking with the `imsharpen` function. Display the original and the resulting 4 sharpened images in a 3x2 grid using the subplot command with each image magnified to 200x200 pixels using the truesize command.

d) Zoom in on the right eye in the image by creating a 100x100 pixel sub-image from each of the original and sharpened images. The top left corner of the sub image is at location (150, 350) and the bottom right corner is at the location (249, 449). Display the resulting 5 images in a 3x2 grid using the subplot command with each image magnified to 200x200 pixels using the truesize command. Compare the sharpened images with the original image and comment on the relative effect of each sharpening.

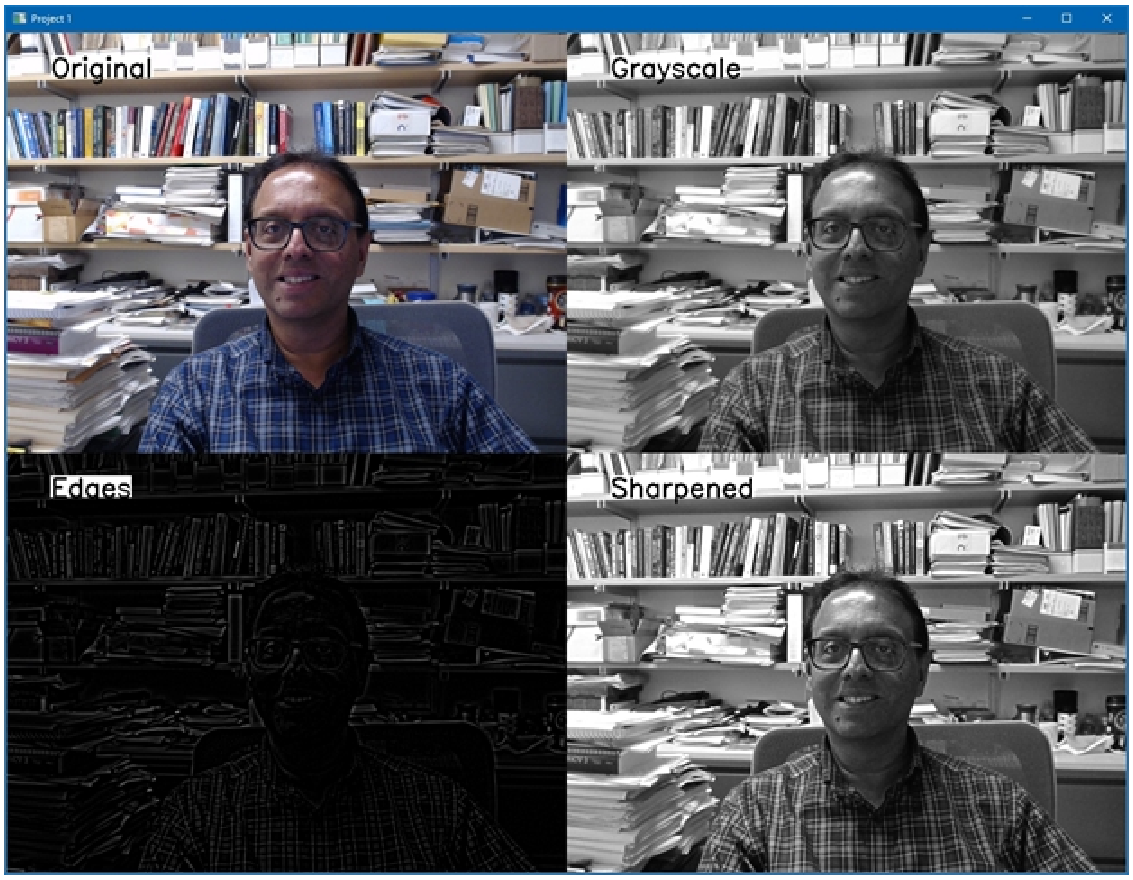
Project 1 - Task 4

OpenCV and Basic Video Functions and Processing

The purpose of this project is to get familiar with the video functions and processing in OpenCV. Review the following basic functions in OpenCV before working on this project:

`VideoCapture`	`Open a camera for video capturing`
`imshow`	`Displays an image in the specified window`
`cvtColor`	`Converts an image from one color space to another`
`GaussianBlur`	`Blurs an image using a Gaussian filter`
`addWeighted`	`Calculates the weighted sum of two arrays`
`putText`	`Draws a text string`

Using above OpenCV functions, process the video stream from a webcam to show 4 simultaneous videos: original color video, grayscale video, video showing edges in the frame, and sharpened video. The final video should look like following:



The resulting video should:

1. Show all 4 videos simultaneously in one window.
2. The video size set to 640 x 480.
3. Top left frame has original color video.
4. Top right frame is the grayscale version of video.
5. Bottom left frame shows the edges in the video.
6. Bottom right frame shows the sharpened video.
7. Put title text on each video frame.

Tips:

- Use the provided VideoFilteringExample code as a template for your program.
- Use the Gaussian filter with a kernel size of 15x15 to blur the image for extracting edges.
- Use addWeighted function to add/subtract image matrices with selected weights.
- Sharpen the image by adding 70% of edges back into original video frame.
- Make sure to convert each video frame to RGB color space before putting together the concatenated frames in the single window.

Report:

- Write your code in the Jupyter Notebook.
- Comment your code sufficiently.
- Print a preview of the program with its output and attach it with this report.
- Capture a separate video with each of group members individually in the video. This can be done by running the program developed by the group on each of the group member’s computer and saving screenshots.
- Save a couple of screenshots of that video and attach it with the report.

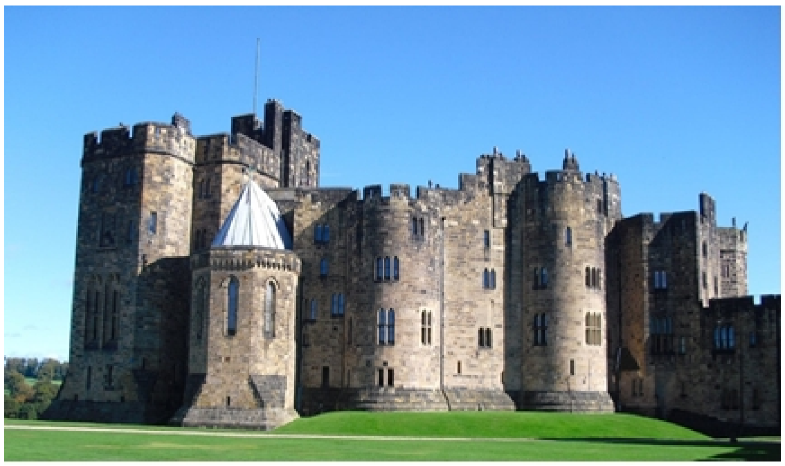
Project 1 - Task 3

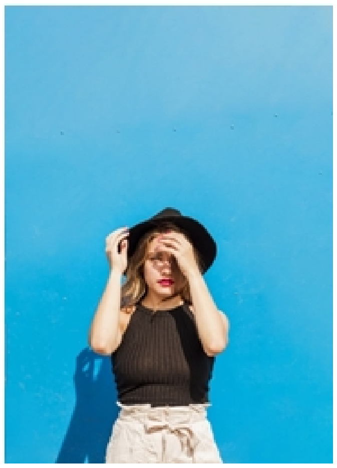
OpenCV and Basic Image Functions and Processing

The purpose of this project is to get familiar with the image functions and processing in OpenCV. Review the following basic functions in OpenCV before working on this project:

`imread`	Loads an image from a file
`imshow`	Displays an image in the specified window
`imwrite`	Saves an image to a specified file
`cvtColor`	Converts an image from one color space to another
`inRange`	Checks if array elements lie between the elements of two other arrays
`bitwise_not`	Inverts every bit of an array
`bitwise_and`	Computes bitwise conjunction of two arrays
`add`	Calculates the per-element sum of two arrays or an array and a scalar
`GaussianBlur`	Blurs an image using a Gaussian filter

Using above OpenCV functions, combine the “AlnwickCastle.jpg” image with the “HatGirl.jpg” image shown below:

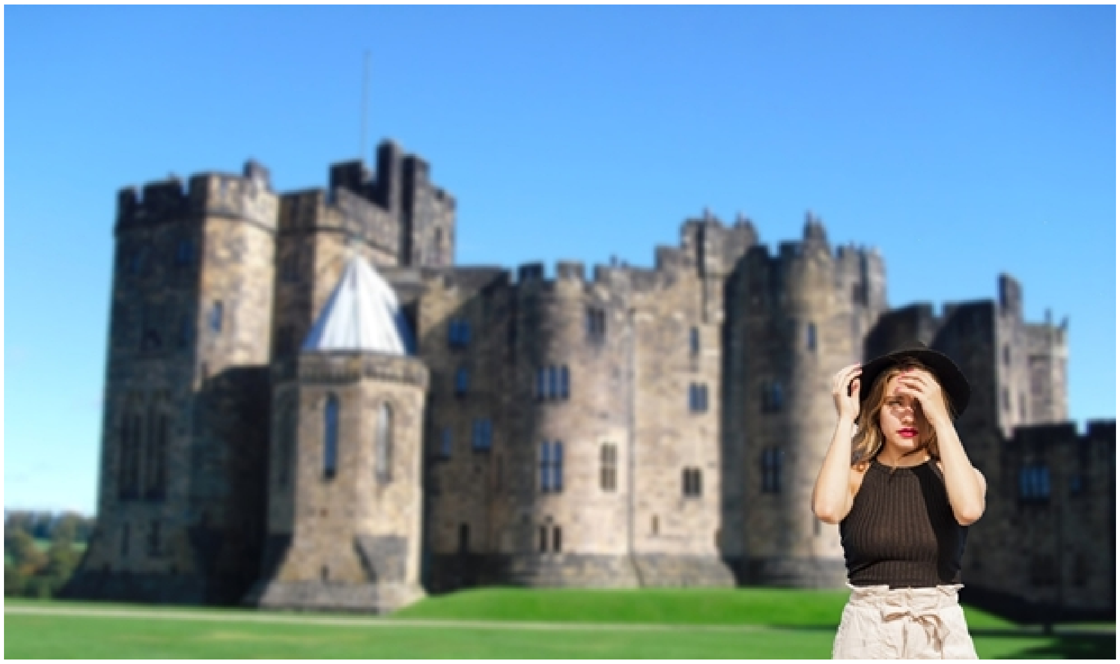




The combined image should:

1. Isolate the girl in the "HatGirl.jpg" image.
2. Blur the castle image in the "AlnwickCastle.jpg" image.
3. Locate the girl in the foreground at the bottom right of the background castle image.
4. Do not change the sizes of both images in the resulting image.
5. Save the resulting image to a jpg file.

The resulting image should look like the following:



Tips:

- Use color isolation method to create a mask to separate the image of the girl.
- Use the Gaussian filter to blur the image. Choose a kernel size to get the blur as close to the one shown in the resulting image above.
- Since the two images are of different sizes and we are not allowed to change the size of images, we cannot directly add the two images as the corresponding matrices are not of the same size.
- Create a region of interest (ROI) at the bottom of the castle image which has the same size as of the girl image and use that portion of the castle image to perform math operations on the same sized matrices to combine the two images using masks.
- Once the two images are combined in ROI, overwrite the ROI in the castle image to obtain the resulting image.

Report:

- Write your code in the Jupyter Notebook and plot all intermediate images inline in the notebook using matplotlib commands.
- Comment your code sufficiently.
- Print a preview of the program with its output and attach it with this report.
- Print the resulting jpg image saved and attach it with this report.