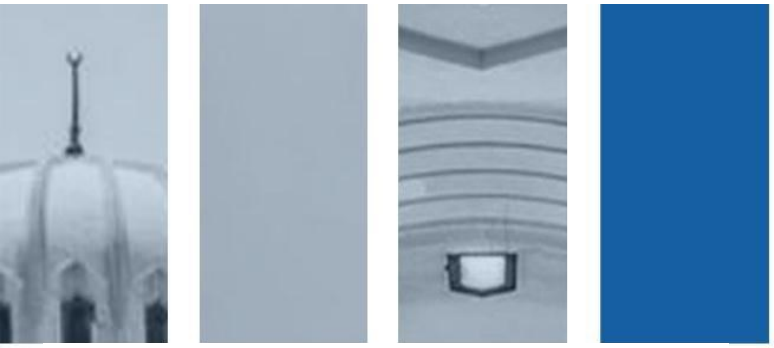


Multimedia Project

Program: Mainstream.

Course: CSE 412

Selected Topics in Computer Engineering – 2nd Semester 2020/2021

Team Names:

Mohammed Obada Bahaa Mohammed Sarsar	(16W0061)
Mostafa Ashraf Mohammed	(16T0107)
Ali Sayed Mohammed Ali	(1600832)
Ghada Ragab Abdelnabi	(1600953)
Mariam Hamdy Abdelmaksoud Afify	(1601367)
Mariam Atef Shafik	(1601372)
Mary Malak Labib Eskander	(1601069)
Mai AboElmaaty Mohame	(1401469)

Contents

1). Introduction.

FOR most video databases, browsing as a means of retrieval is impractical, and query-based searching is required. Queries are often expressed as keywords, requiring the videos to be tagged.

In view of the ever-increasing size of video databases, the assumption of an appropriate and complete set of tags might be invalid, and content-based video retrieval techniques become vital.

Video is an example of multimedia information, which contains a huge set of raw data, richer content information, and made up of continuous still images. This makes the retrieval of videos more tedious and challenging.

The content-based video retrieval has attracted the users as the use of multimedia data has been increased drastically in day-to-day life compared to text-based search. Text-based search or retrieval is good when the database is small and limited.

But when it comes to huge and larger databases or data warehouses, content-based retrieval plays a vital role. Hence making the system more robust and reliable as well as automatized is necessary for the domain of the CBVR system.

The CBVR basically retrieves a related video from the database based on visual contents like color, texture, edge, shape, and motion.

2). Detailed Project Description.

* We have data consisting of images set and videos set as metadata and want to use content-based retrieval algorithms to find the data (images or videos) that is required from our metadata

If the query data was an image, we use 3 algorithms to find it and return all similar images from our database:

1-Compare by Mean Color:

In this method, we compare the mean color value for the query image and the mean values of the images that are store in our database and return the index of all images that approximately equal to the query image.

2. Compare by Histogram:

In this method, we compare the Histogram values for the query image and Histograms of the images that are store in our database and return the index of all images that approximately equal to the query image.

3. Color layout algorithm.

The third algorithm for images is the color layout algorithm which is a global color feature, it divides the whole image into sub-blocks then extracts features from each sub-block or divides them into regions based on features as this process is called segmentation.
The color layout is a very compact and resolution-invariant representation of color for high-speed image retrieval, and it has been designed to efficiently represent the spatial distribution of colors. This feature can be used for a wide variety of similarity-based retrieval, content filtering, and visualization. It is especially useful for spatial structure-based retrieval applications. This descriptor is obtained by applying the DCT transformation on a 2-D array of local representative colors in Y or Cb or Cr color space. The functionalities of the color layout are basically the matching:
– Image-to-image matching
– Video clip-to-video clip matching
The color layout algorithm is one of the most precise and fast color descriptors.
When we store the features of each image on the database by using this algorithm, we can also compare the input image after obtaining its features and get similar images for the input image.
The implementation of the algorithm is just dividing the whole image into sub-blocks then calculating the histogram of each sub-block and applying the histogram algorithm on it by the comparing method of histogram algorithm.

* And if the query data was a video, we use one algorithm to find it and return all similar videos from our database:

4. naive video similarity:

In this algorithm, we compare the features of keyframes in the query video, which in our case is the histogram of the keyframe to the keyframes features for all videos that are stored in our database.

• A video similarity measure that is based on the percentage of visually similar frames between two sequences.

• A naive way to compute such a measure is to first find the total number of frames from each video

sequence that has at least one visually similar frame in the other sequence,

and then compute the ratio of this number to the overall total number of frames.

• The number of comparisons can be reduced by selecting frames in each video sequence that are

similar to a set of predefined random feature vectors common to all video sequences

3). Beneficiaries of The Project.

This project for anyone who wants to search for something by image and gets images similar to one he has, or has part of the video and wants to search for similar videos.

For example, if he wants a photo for a certain flower and has its image could find other flower images by this project or if he has a scene of a movie and needs the whole movie.

4). Detailed Analysis.

We have images and videos stored on the database and the user enters an input image or video then the system searches for a similar image or video for the input and retrieves them to the user.

For the image retrieval part, we have three algorithms: Mean color, histogram, and color layout

Mean color: The algorithm calculates the average of the red color and the average for the green color and the average of the blue color that forms the whole image as it calculates the average for the three channels RGB.

Histogram: The algorithm is a graph for the frequency of pixels intensity in the image, the x-axis of the histogram shows the range of pixels value, and the y-axis shows the frequency of each color and the number of colors occurrence.

Color layout algorithm.
The third algorithm for images is the color layout algorithm which is global color features, it divides the whole image into sub-blocks then extracts features from each sub-block or divides them into regions based on features as this process is called segmentation.
The color layout is a very compact and resolution-invariant representation of color for high-speed image retrieval, and it has been designed to efficiently represent the spatial distribution of colors. This feature can be used for a wide variety of similarity-based retrieval, content filtering, and visualization. It is especially useful for spatial structure-based retrieval applications. This descriptor is obtained by applying the DCT transformation on a 2-D array of local representative colors in Y or Cb or Cr color space. The functionalities of the color layout are basically the matching:
– Image-to-image matching
– Video clip-to-video clip matching
The color layout algorithm is one of the most precise and fast color descriptors.
When we store the features of each image on the database by using this algorithm, we can also compare the input image after obtaining its features and get similar images for the input image.
The implementation of the algorithm is just dividing the whole image into sub-blocks then calculating the histogram of each sub-block and applying the histogram algorithm on it by the comparing method of histogram algorithm.

5). Detailed Description of The Adopted Techniques.

a- Mean Color algorithm

The first algorithm for images is the Mean Color algorithm which calculates the average of the red color and the average for the green color and the average of the blue color that forms the whole image as it calculates the average for the three channels RGB, from here we able to extract color features from the images and compare the color feature of the different images, it facilitates the content-based image retrieval process, it might be the similarity between some images for their RGB distribution so a threshold number will be put into consideration for the calculations to retrieve a range of images.

The mechanism of the mean color algorithm for color feature extraction is calculating the mean color for RGB to the query image from the user and each image in the database and store it in the database as a color feature field, the first step of the algorithm is a function called “calcMeanImage()” which takes the image path as an argument and read it then calculates the

mean by using the built-in function “mean”, then return the 3 values of the average for the three channels RGB in form of a list.

The second step is to compare the query image and each image in the database by a function called “compareMean()” which takes the query image from the user and index from the database as arguments, we iterate over the images on the database and put “if” condition for the three-average values of the RGB and use the threshold value from 0.9 to 1.1, then it will return a list containing the similar images from the database to the query image.

b- Histogram algorithm.

The second algorithm for images is the Histogram algorithm which is a graph for the frequency of pixels intensity in the image, the x-axis of the histogram shows the range of pixels values, and the y-axis shows the frequency of each color and the number of color occurrence, the algorithm is most commonly used color features representation as we can predict about an image by just looking at its histogram, we can compare two histograms together to get all the similar images for a given image, so it facilitates the feature extraction process for content-based image retrieval system.

The mechanism of the histogram algorithm for image feature extraction is calculating the histogram for each image has stored in the database and the histogram for the query image that the user has entered, we need to take another step which is normalization as it is a common technique that is used to enhance fine detail with an image as it converts the range of pixels intensity on the x-axis to a specific range we have chosen to facilitate the calculations with the histogram, then compare between the two histograms.

The implementation of the first part is a function called “calcHistogram()” takes the path of the image that stored in the database as an argument then simply reads the image path then calculate the histogram of this image and normalize it by using built-in functions in OpenCV library which are “imread()”, “calcHist()”, “normalize()”, then return the histogram of the image in a list form and store this list in the database as feature field.

The second step is another function whose name is “comparehistogram()” which is take the query image from the user and an index from the database, first, it compares the query image with an image from the database with its index by the correlation comparison technique between the two histograms with an iterative way inside a “for” loop that iterates over the images in the database by its index by using “compareHist()” function, then we sorted the result from the comparison in descending order and append them in an array in which the most similar image will be displayed till the lest similar image, then return the result list of similar images to the query image.

c- Color layout algorithm.

The third algorithm for images is the color layout algorithm which is global color features, it divides the whole image into sub-blocks then extracts features from each sub-block or divides them into regions based on features as this process is called segmentation.

The color layout is a very compact and resolution-invariant representation of color for high-speed image retrieval, and it has been designed to efficiently represent the spatial distribution of colors. This feature can be used for a wide variety of similarity-based retrieval, content filtering, and visualization. It is especially useful for spatial structure-based retrieval applications. This descriptor is obtained by applying the DCT transformation on a 2-D array of local representative colors in Y or Cb or Cr color space. The functionalities of the color layout are basically the matching:

– Image-to-image matching

– Video clip-to-video clip matching

The color layout algorithm is one of the most precise and fast color descriptors.

When we store the features of each image on the database by using this algorithm, we can also compare the input image after obtaining its features and get similar images for the input image.

The implementation of the algorithm is just dividing the whole image into sub-blocks then calculating the histogram of each sub-block and applying the histogram algorithm on it by the comparing method of histogram algorithm.

d- Naïve Video Similarity algorithm.

One definition of a video similarity measure suitable for these applications is in terms of the percentage of visually similar frames between two video sequences. This is the video similarity measure used in this project.

There are other similarity measures proposed in the literature and some of them are reviewed in this report. No matter which measure is used, the major challenge is how to efficiently perform the measurement. As video is a complex data type, many of the proposed similarity measures are computationally intensive. On the other hand, for every new video added to the database or a query video presented by a user, similarity measurements need to be performed with possibly millions of entries in the database. Thus, it is imperative to develop fast methods in computing similarity measurements for database applications.

Let X and Y be two video sequences. The number of frames in video X that have at least one

visually similar frame in Y is represented by Px∈X1{y∈Y : d(x,y)≤²}, where 1a is the indicator function with 1a= 1 if A is not empty, and zero otherwise. The Naive Video Similarity between X and Y , NVS(X, Y; ²), can thus be defined as follows:

$$\mathrm{nvs}(X, Y; \epsilon) \triangleq \frac{\sum_{x \in X} 1_{\{y \in Y:\, d(x,y) \leq \epsilon\}} + \sum_{y \in Y} 1_{\{x \in X:\, d(y,x) \leq \epsilon\}}}{|X| + |Y|}, \qquad (1)$$

where | · | denotes the cardinality of a set, or in our case the number of frames in a given video.

If every frame in video X has a similar match in Y and vice versa, NVS(X, Y; ²) = 1. If X and Y share

no similar frames at all, NVS(X, Y; ²) = 0.

6). Task Breakdown Structure.

We divide the whole project into four tasks which are:

The first part is image features extraction with its three algorithms which are Mean color, histogram, and color layout algorithm.

The second part is video features extraction with one algorithm which is naive video similarity.

The third part is a database.

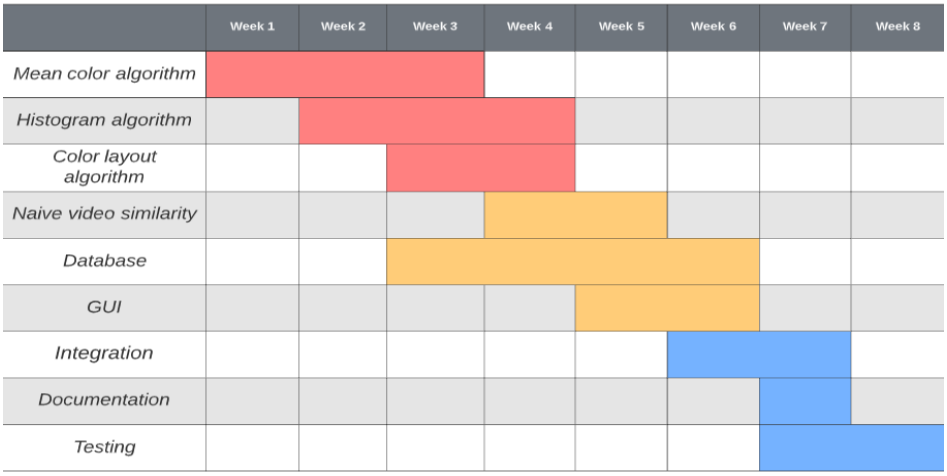
The fourth part is the GUI which is a desktop application using python.

Then the integration of the four parts.

All of us contributed to writing the documentation.

We work together on each part of the project.

7). Time Plan of The Project.



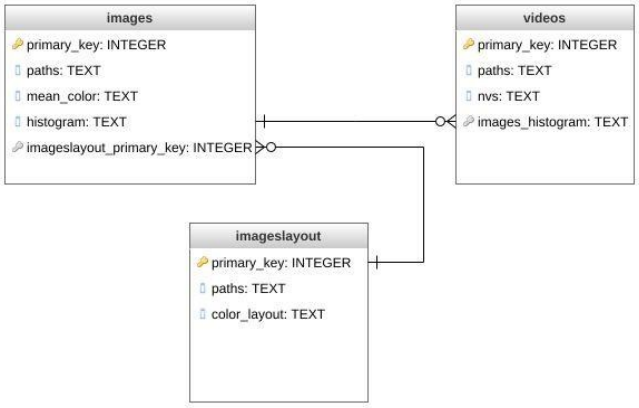
"Fig.1: Time plan of the project"

The team divided into 2 subgroups which are:

-Mariam Atef, Mary Malak, Mariam Hamdy, and Mostafa Ashraf worked on image algorithms + GUI + SW design +readme + documentation.

-Ali Said, Ghada Ragab, Mai Maaty, and Mohamed Obada worked on the video algorithm + DB + UI to backend connection + presentation + documentation.

8). Multimedia Database Design.



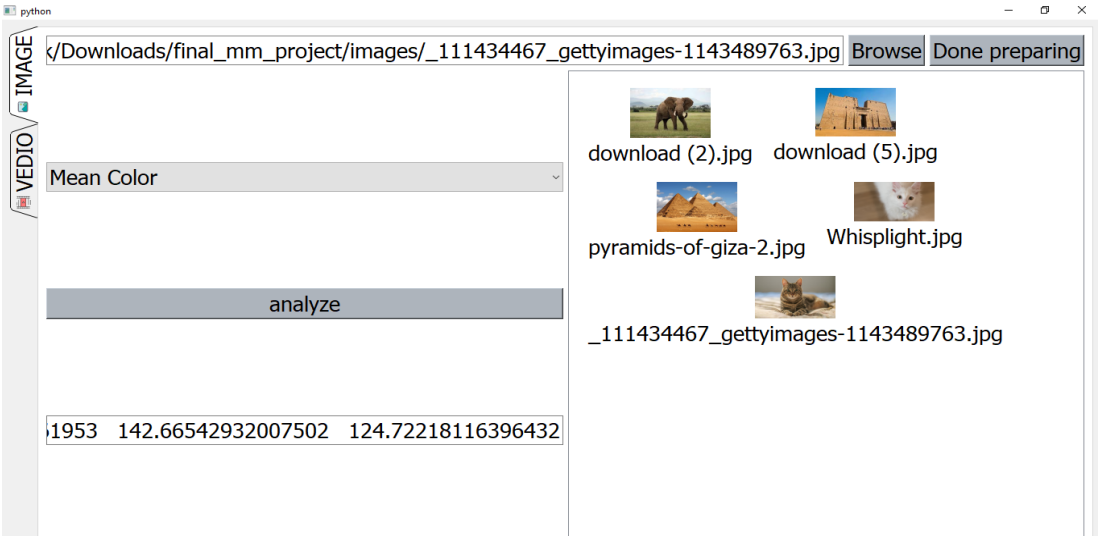
"Fig.2: Database design"

9). System Design.

"Fig.3: Class diagram"

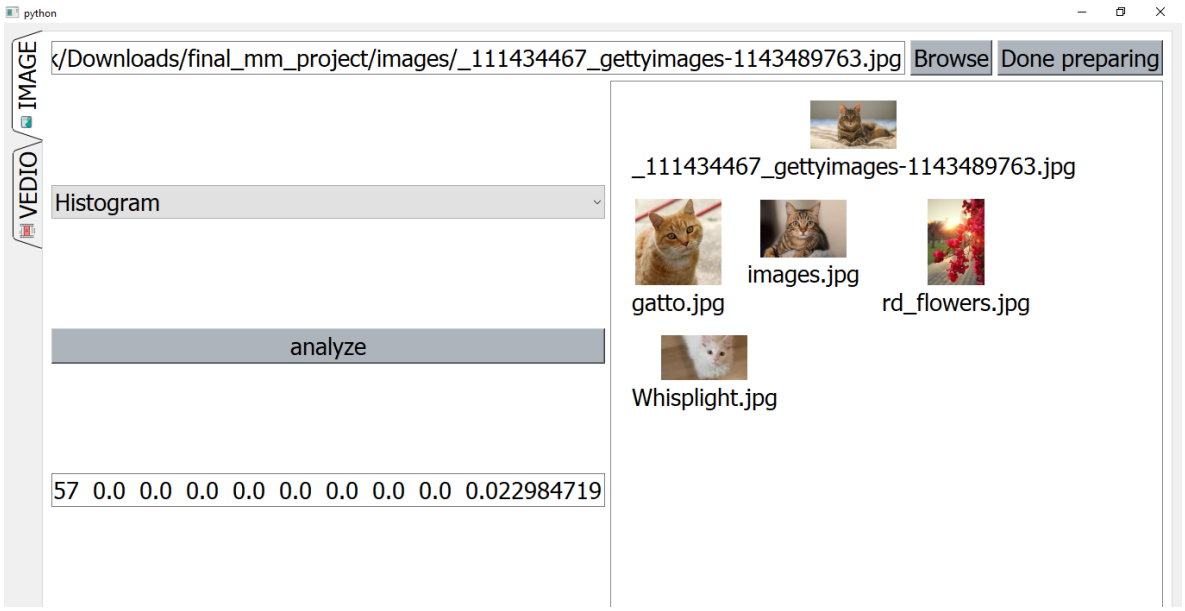
10). Testing Scenarios and Results.

a-First example of pictures using the mean color algorithm.



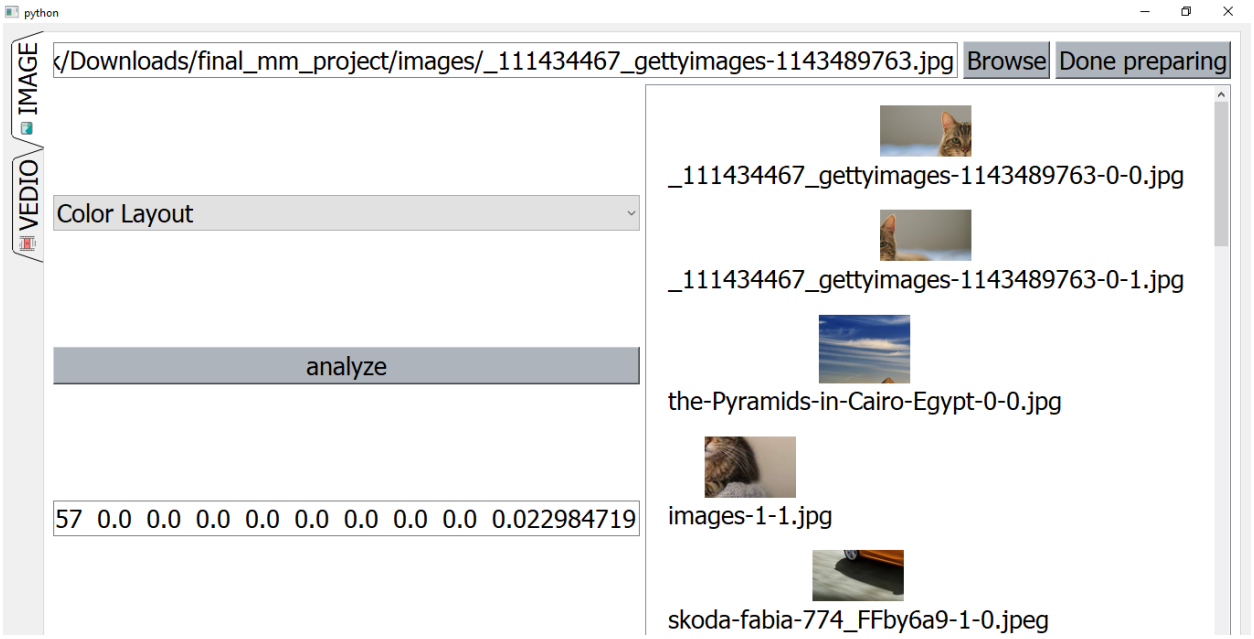
"Fig.4: Mean color example"

b-First example of pictures using histogram algorithm.



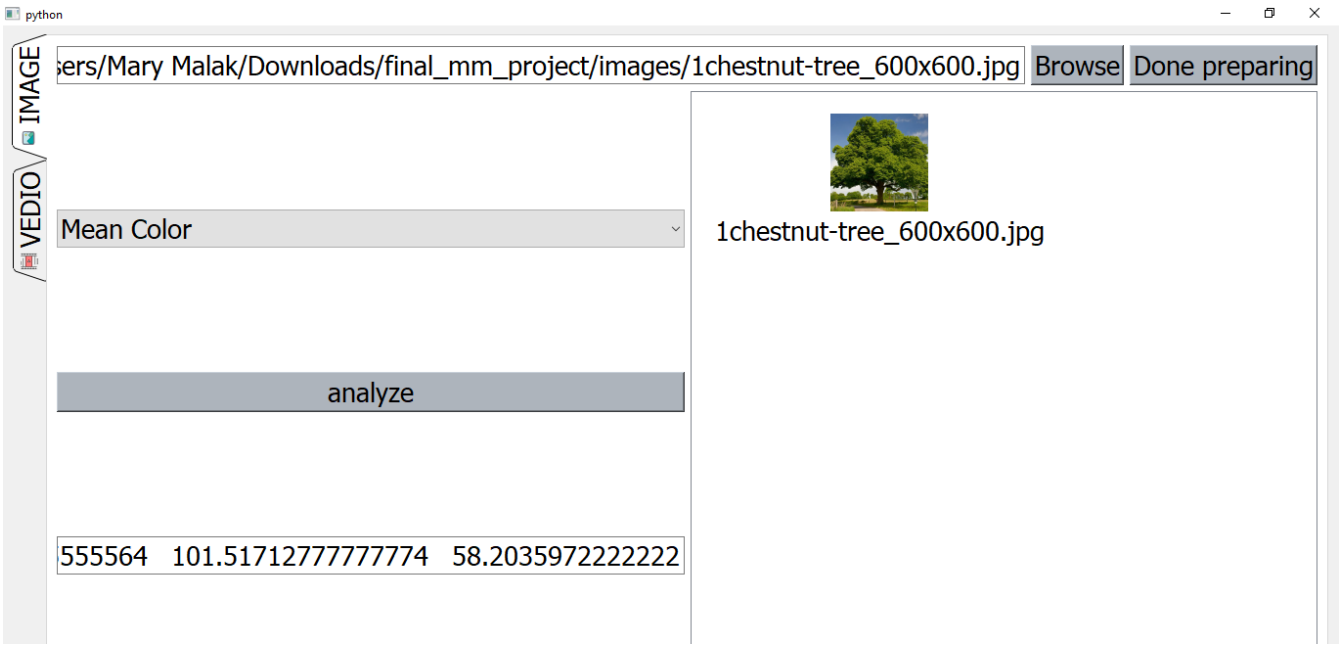
"Fig.5: Histogram example"

c-First example of pictures using color layout algorithm.



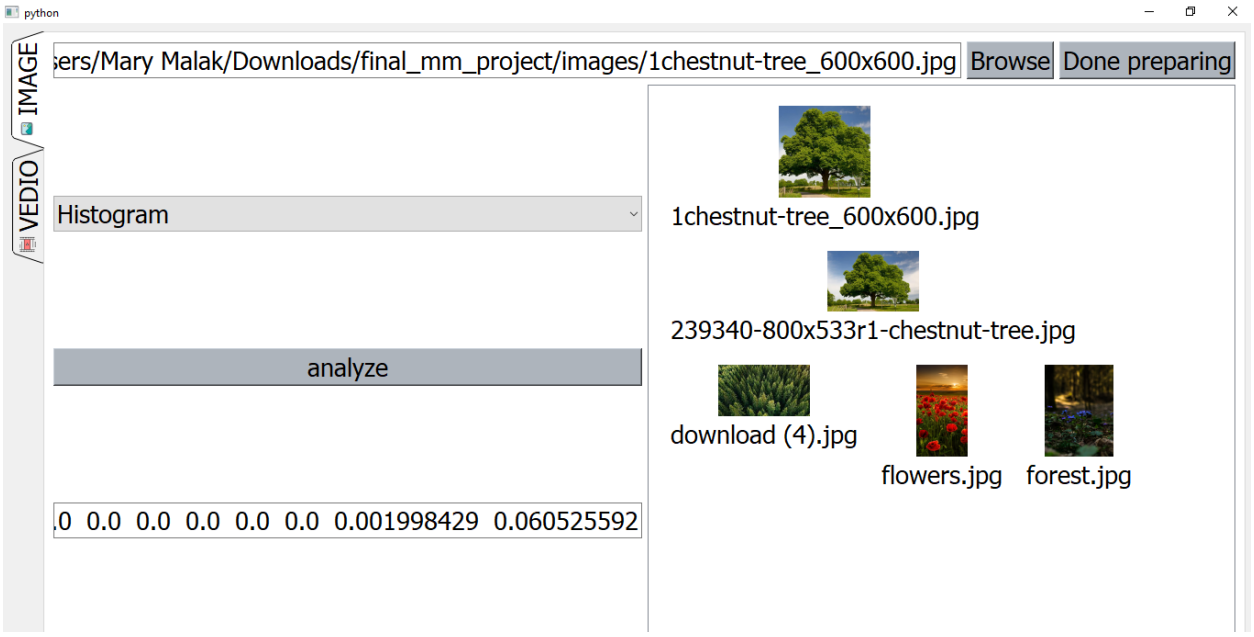
"Fig.6: Color layout example"

d-Second example of pictures using the mean color algorithm.



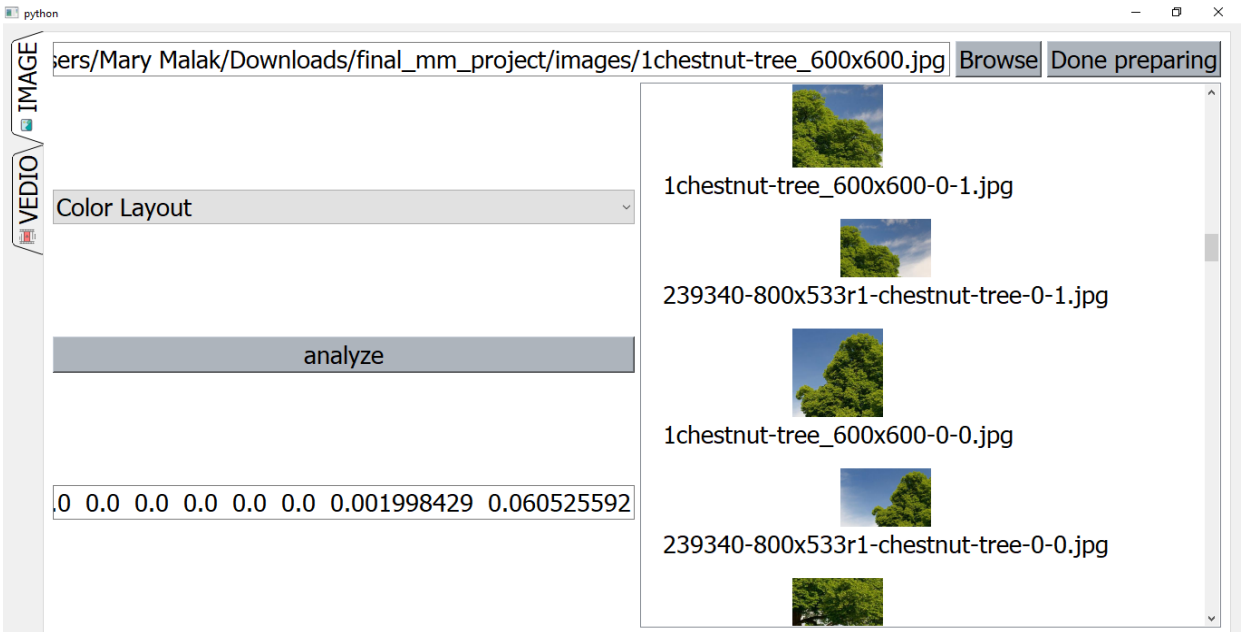
"Fig.7: Mean color example"

e-Second example of pictures using histogram algorithm.



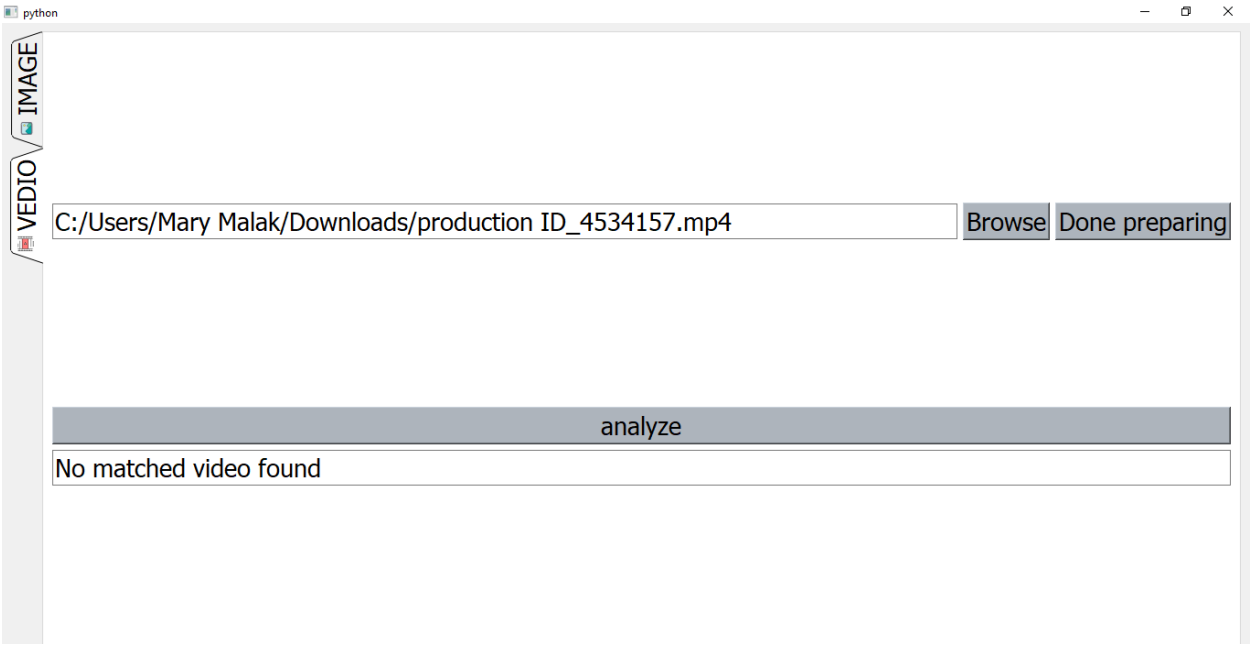
"Fig.8: Histogram example"

f- Second example of pictures using color layout algorithm.



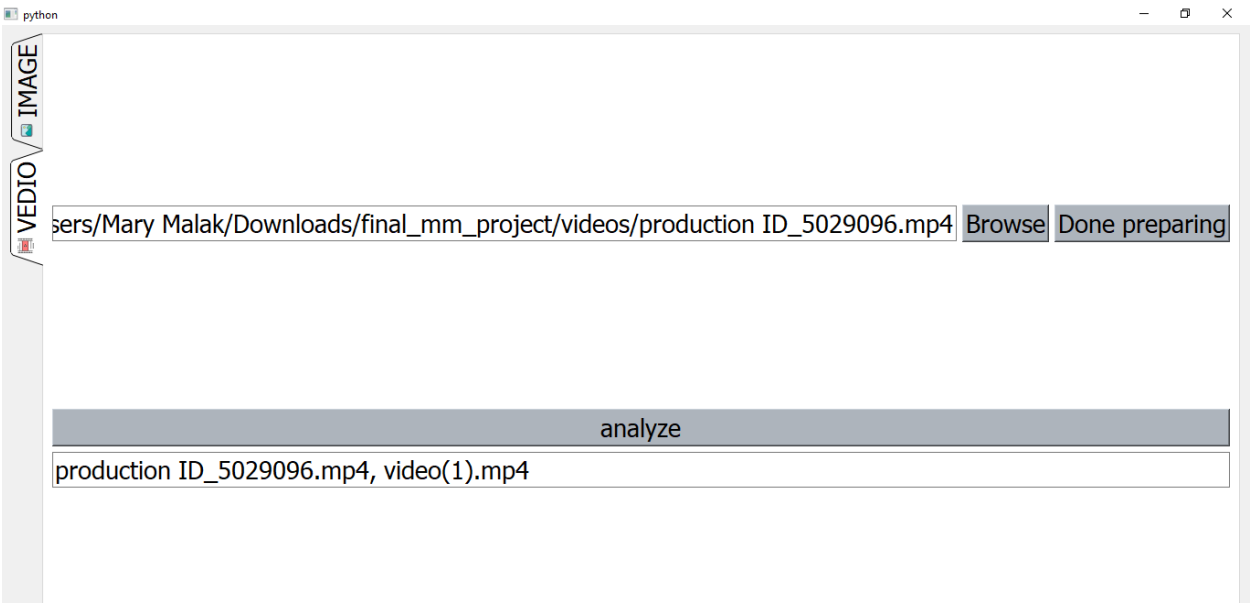
"Fig.9: Color layout example"

g-First example of videos using naïve video similarity.



"Fig.10: Video searching example."

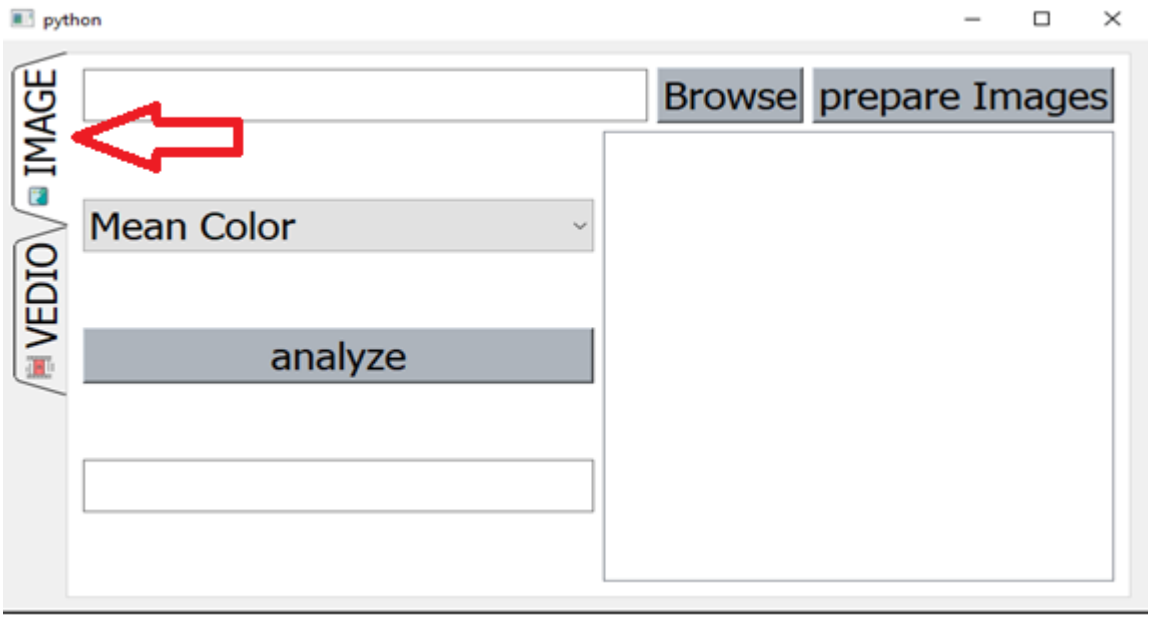
h- Second example of videos using naïve video similarity.



"Fig.11: Video searching example."

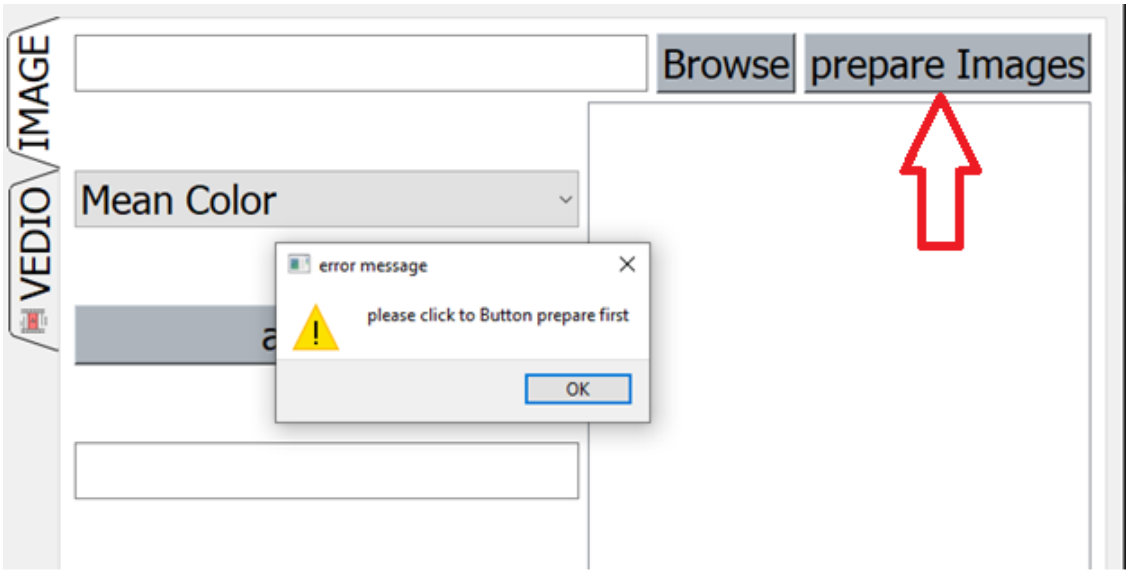
11). End-User Guide.

a- Open the application (.exe) and choose the “image” tap.



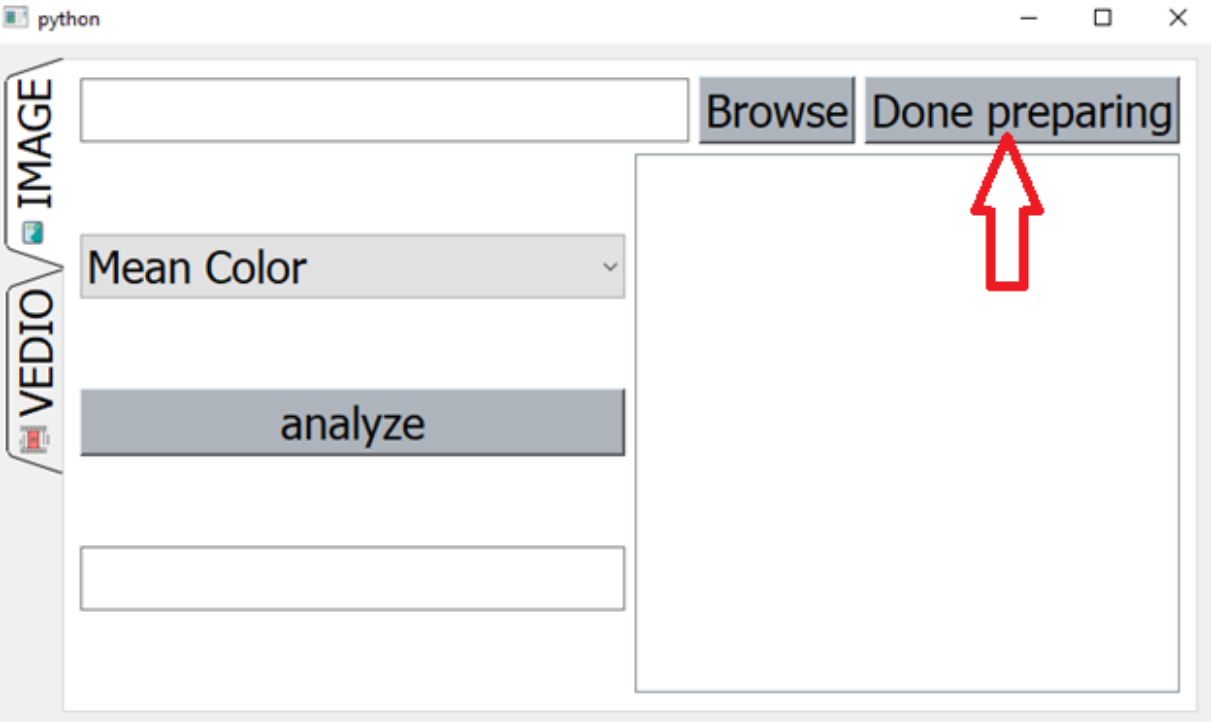
“Fig.12: Main GUI Interface of Images”

b- we should click “prepare image” to prepare a database for images.



“Fig.13: Searching Operation Failure.”

*Must click “prepare images” before clicking “Analyze”.



“Fig.14: Preparing Database Success.”

c- Showing Database Tables

The first table consists of 3 columns 1st column containing the relative path of each image we have in the images folder the 2nd column containing the features of Mean color for each image the 3rd column containing the features of the histogram for each image.

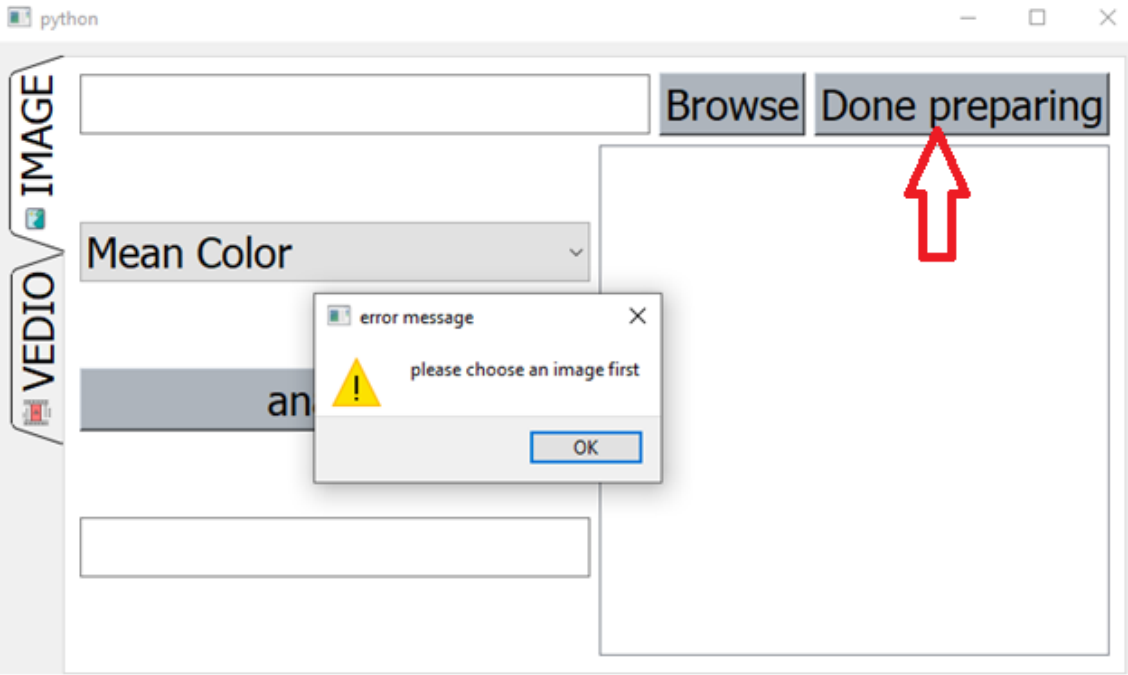
The second table contains the relative paths of the image and its color layout.

	paths text	mean_color text	histogram text
1	1200px-A...	177.161855680...	0.007826008,0...
2	1chestnu...	87.2232555555...	0.4172208,0.0...
3	239340-8...	107.424737335...	0.30017093,0....
4	7GIUFPIJ...	150.611876929...	0.07565957,0....
5	Abu-Simb...	176.252155844...	0.043621194,0...
6	All_Gizah...	177.161855680...	0.007826008,0...
7	Ash_Tree...	111.901147998...	0.5915553,0.0...
8	beautiful-...	90.4528592377...	0.0882406,0.0...
9	black-do...	142.441300450...	0.41266847,0....
10	Capture.p...	194.813704769...	0.025123665,0...
11	cat-4102...	167.520010544...	0.12906654,0....
12	cat.jpg	189.196345270...	0.0064625302,...
13	doge.jpg	204.665429522...	0.053456545,0...
14	downloa...	67.2602151813...	0.6463612,0.0...
15	downloa...	151.783050173...	0.06008079,0....
16	downloa...	142.207337951...	0.18312427,8....
17	downloa...	53.9558531746...	0.6303648,0.0...
18	downloa...	154.396780134...	0.06726084,0....
19	downloa...	162.541997019...	0.24251269,0....
20	downloa...	196.608236808...	0.000723252,0...

	paths text	color_layout text
1	1200px-A...	0.001401419,0.0...
2	1200px-A...	0.0007378018,0....
3	1200px-A...	0.011878384,0.0...
4	1200px-A...	0.013214577,0.0...
5	1chestnu...	0.28101826,0.00...
6	1chestnu...	0.14114729,0.0,0...
7	1chestnu...	0.67234075,0.01...
8	1chestnu...	0.3699336,0.036...
9	239340-8...	0.17033303,0.0,0...
10	cat-4102...	0.3302694,0.092...
11	239340-8...	0.07975033,3.80...
12	239340-8...	0.4656498,0.017...
13	239340-8...	0.24885833,0.01...
14	7GIUFPIJ...	0.12446401,0.00...
15	7GIUFPIJ...	0.016713804,0.0...
16	7GIUFPIJ...	0.107773475,0.0...
17	7GIUFPIJ...	0.035429258,0.0...
18	Abu-Simb...	0.00018021892,...
19	Abu-Simb...	0.02670228,0.16...
20	Abu-Simb...	0.0571121240.0

"Fig.15: Image database."

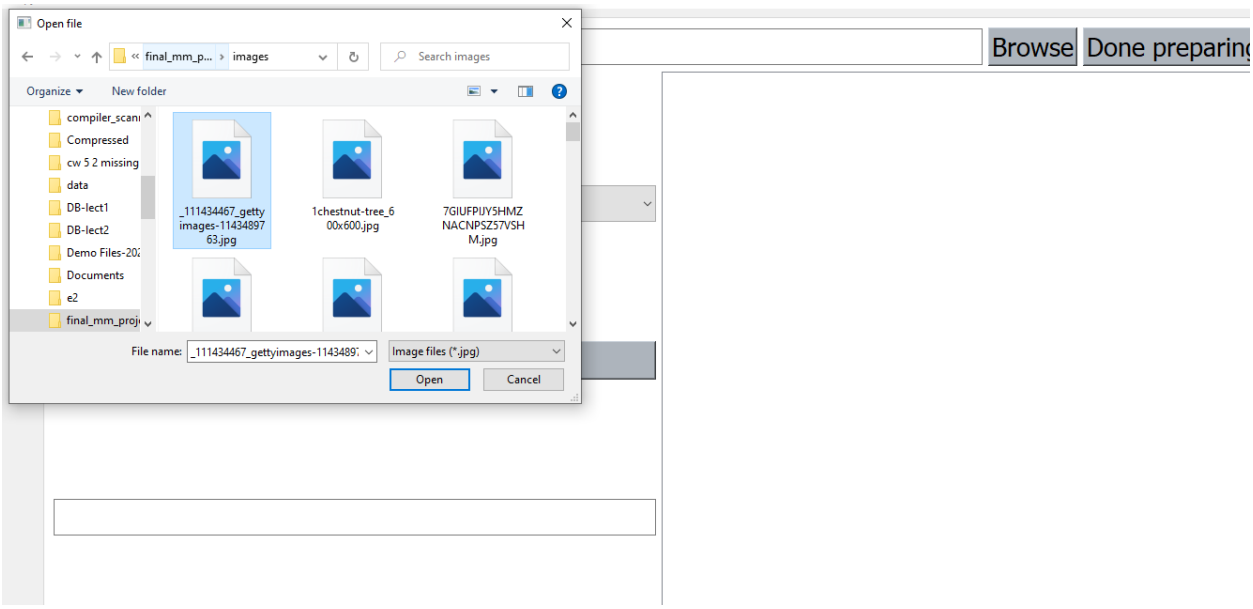
d- Click "Browse" to choose which image we search.



"Fig.16: Analyzing Operation Failure."

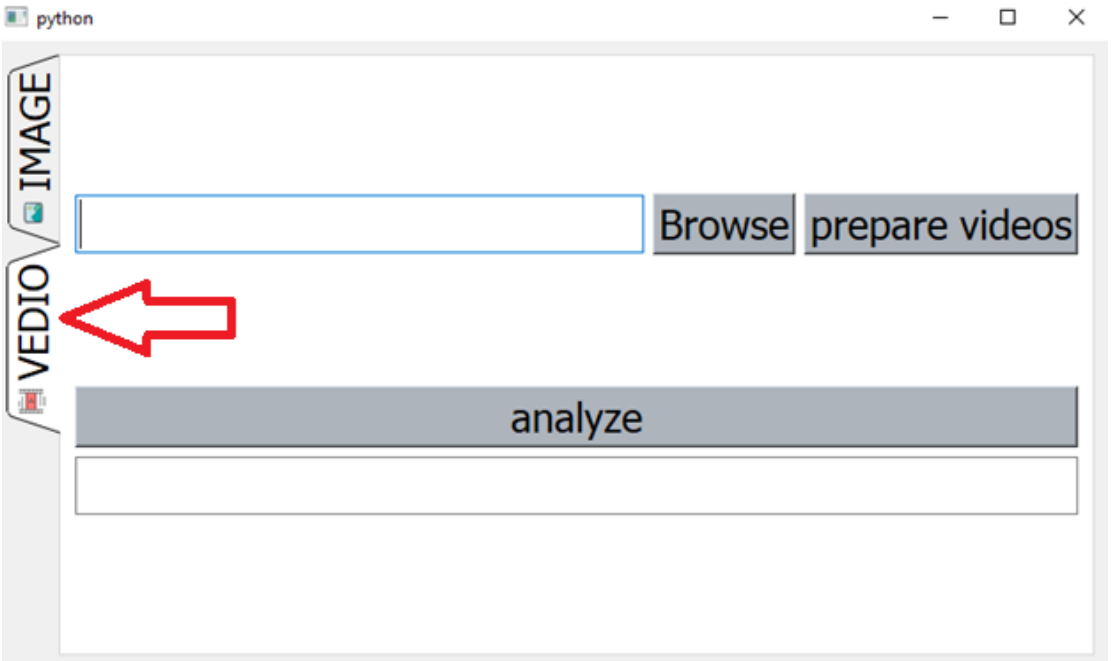
*Must choose an image before clicking "Analyze".

the e-Choose image on our PC.



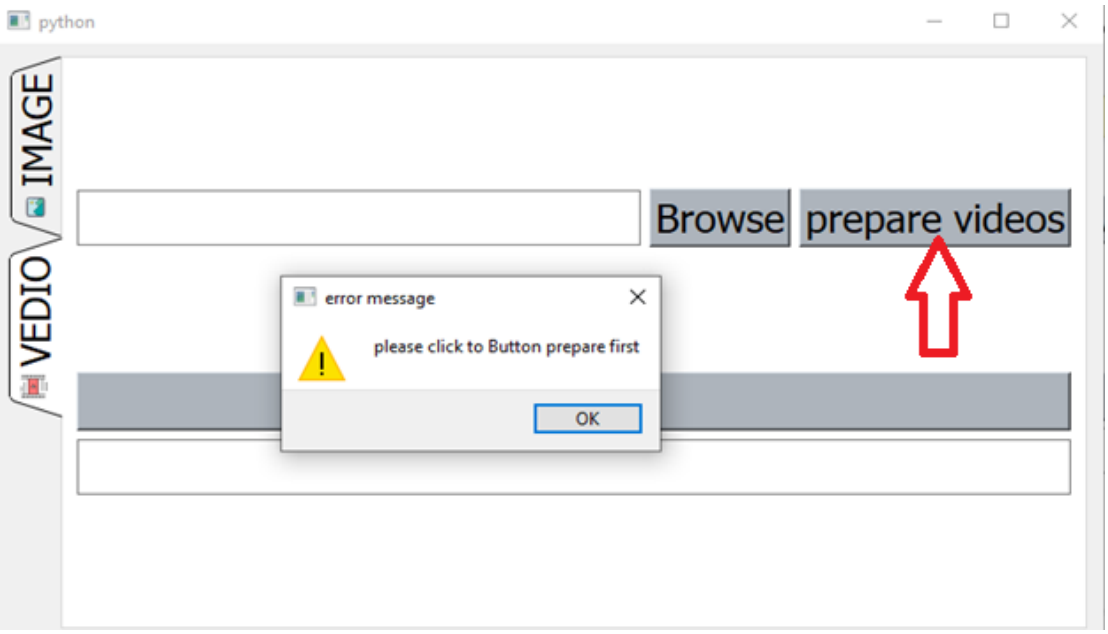
"Fig.17: Choosing Input Image."

f-Click "Video" tap to search in videos.



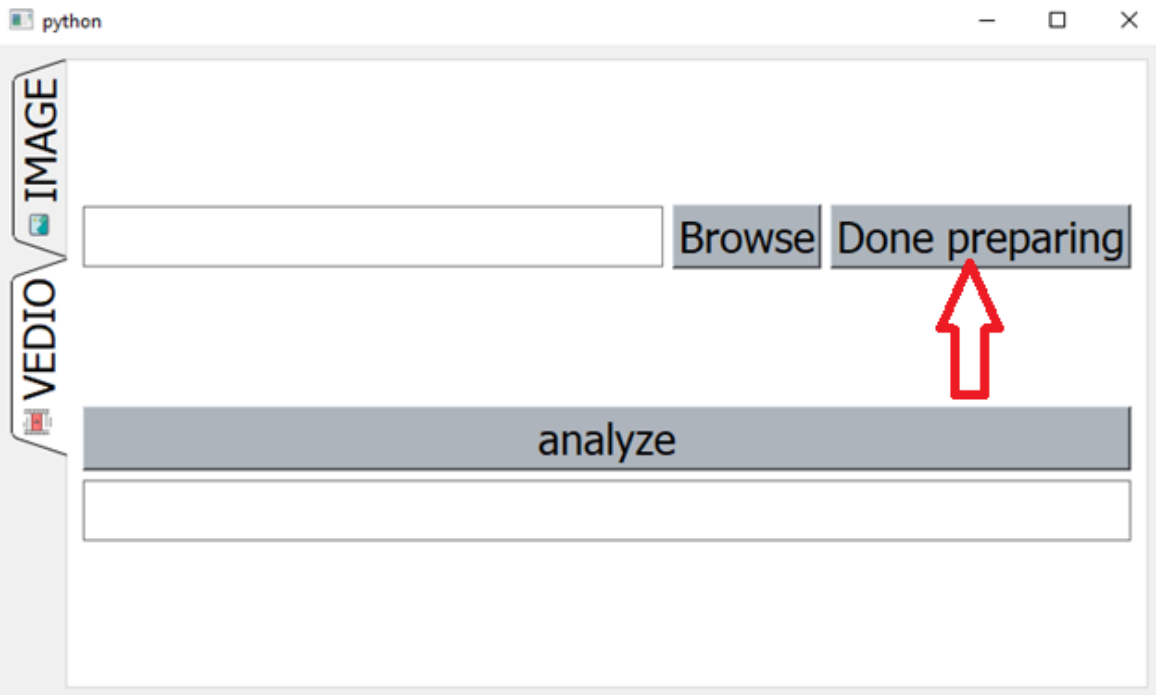
"Fig.18: Main GUI Interface of Videos"

g- we should click to prepare a video to prepare a database for videos.



"Fig.19: Searching Operation Failure."

*must click "prepare images" before clicking "Analyze".

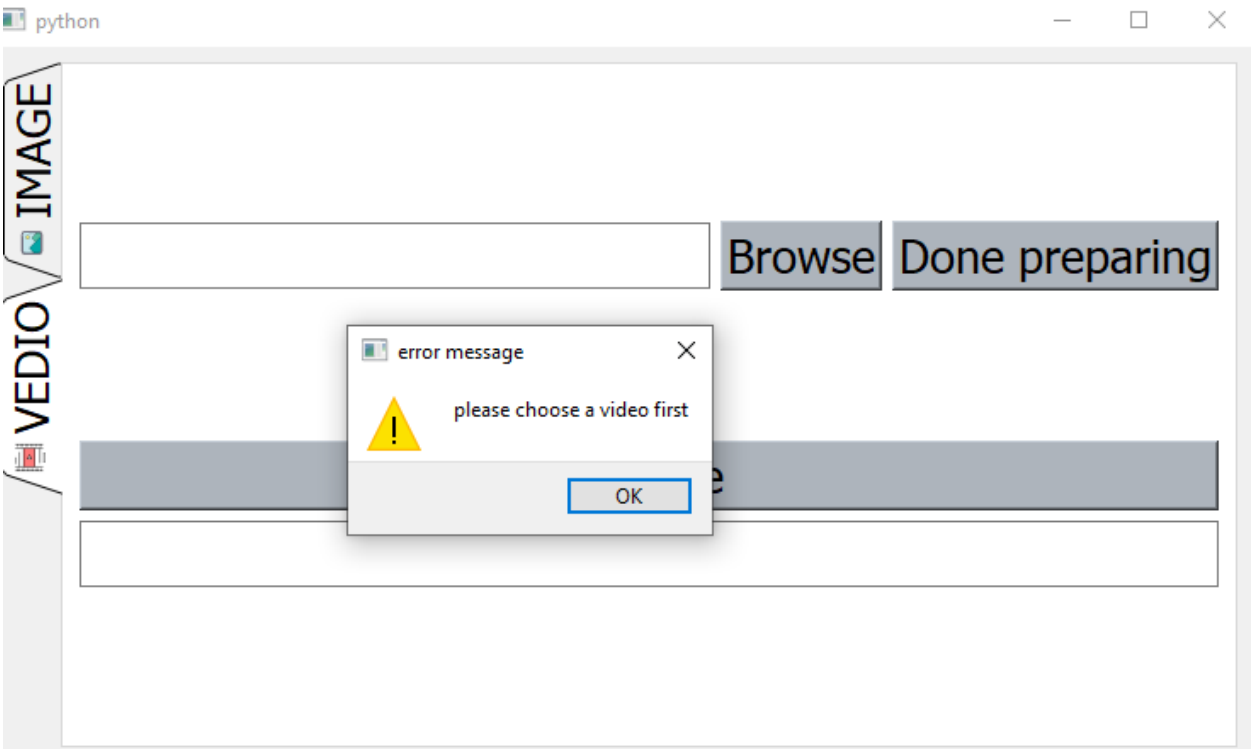


"Fig.20: Preparing Database Success."

	paths text	nvs text
1	productio...	0.0633...
2	video(1)....	0.0633...

"Fig.21: Videos database"

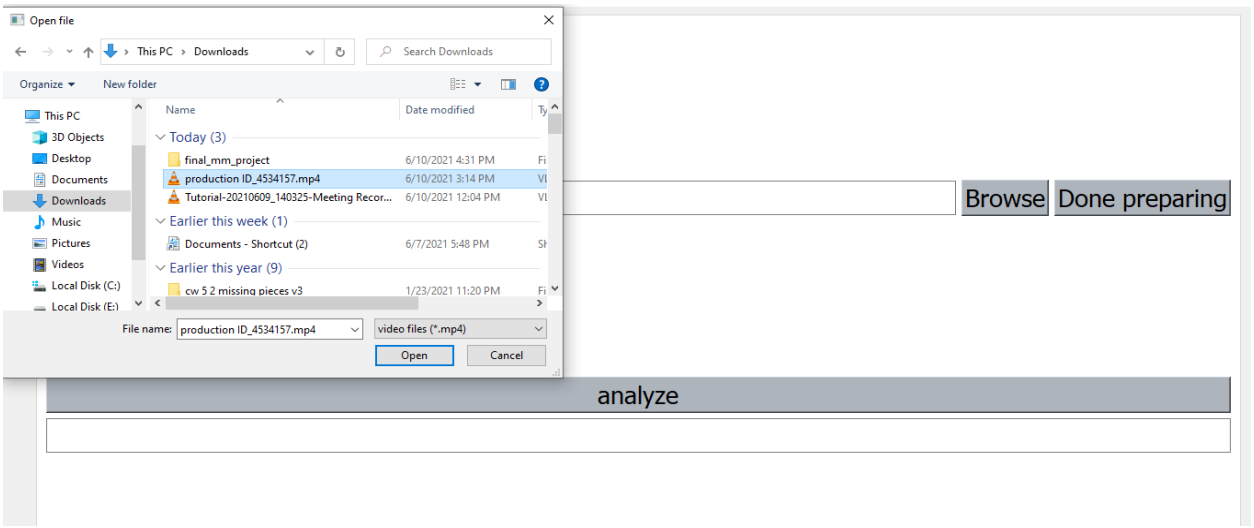
h- Click "Browse" to choose which image we search.



"Fig.22: Analyzing Operation Failure."

*Must choose a video before clicking "Analyze".

i- Choose a video on our PC.



"Fig.23: Choosing Input Video."

12). Conclusion.

Some related work regarding the technologies and applications of content-based image retrieval is presented. Further review of several upgraded CBIR systems is given.

Numerous researches have been accomplished in the CBIR field, it has further to go to supply the users with resources to extract images out of the image or multimedia databases in a really effective method giving input by means of image, text, or drawing.

The main aim of this work is to investigate an approach to using objects for video retrieval and a set of experiments was developed in order to explore some issues related to this.

The focus of this thesis is on general ad hoc retrieval by a hypothetical professional user performing visual queries on a video and image archive.

In this section are discussed the main findings of this research.

Experimental results indicate that object-based search overall outperforms the image-based search for the set of topics used here. However, there are search topics that are not well suited to object-based retrieval.

13). References.

1. M. Mandal, Multimedia Signals and Systems, Kluwer Academic Publishers, 2003.

2. A. Puri, Multimedia Systems, Standards and Networks, Taylor and Francis, 2007.

3. K. R. Rao, Z. S. Bojkovic, and D. A. Milovanovic, Multimedia Communication Systems, Prentice-Hall, 2002.

4. F. Halsall, Multimedia Communications, Addison Wesley, 2001