1. INTRODUCTION

1.1 Image analytics:

Image analytics is the extraction of meaningful information from images; mainly from digital images by means of digital image processing techniques. Image analytics tasks can be as simple as reading bar coded tags or as sophisticated as identifying a person from their face. Image Analytics is a branch of social listening which analyses images, emoji's, memes and other rich media formats to understand their sentiment and how it relates to brand perception.

Image analytics is the automatic algorithmic extraction and logical analysis of information found in image through digital image processing techniques. With an explosion of image data, which makes up about 80 present of all unstructured big data, there is a growing need of analytical systems to interpret images, which is unstructured data to machine readable format. Case in point the use of bar codes and QR codes are simple and popular examples of image analytics that we encounter in our **day-to-day** lives.

Image analysis, which usually returns numeric values and/or graphical information about the image characteristics that are suited for classification, defect detection, or prediction of some of the quality properties of the imaged object. The term image analysis is used when the output is a number or decision, not an image. It is important to note that image analysis is part of a wider field known as image processing, where the main underlying idea is to improve the visual quality of an image and/or to extract useful information or features. The analysis is based on different image properties such as colour, gloss, morphology of the objects, and texture.

Computers are indispensable for the analysis of large amounts of data, for tasks that require complex computation, or for the extraction of quantitative information. On the other hand, the human visual cortex is an excellent image analysis apparatus, especially for extracting higher-level information, and for many applications including medicine, security, and remote sensing human analysts still cannot be replaced by computers. For this reason, many important image analysis tools such as edge detectors and neural networks are inspired by

human visual perception models. Image analytics is the automatic algorithmic extraction and logical analysis of information found in image through digital image processing techniques.

1.2 Data Sets:

A data set is a collection of related data, discrete items of related data that may be accessed individually or in combination or managed as a whole entity. A data set is organized into some type of data structure. In a database, for example, a data set might contain a collection of business data (names, salaries, contact information, sales figures, and so forth). The database itself can be considered a data set, as can bodies of data within it related to a particular type of information, such as sales data for a particular corporate department.

2. LITERATURE REVIEW

Image segmentation is an important step of image processing and it partitions an input image into non-overlapping, homogeneous and connected regions such that "union of any two spatially adjacent regions is not homogenous". Researchers have devised and proposed many techniques for segmentation but no general technique exists, which may be used for all images. However, Keri Wood suggested that good image segmentation should meet the following requirements:

1. Every pixel in the image must belong to a region and each region should be homogeneous with respect to a chosen characteristic, which could be syntactic e.g. colour, intensity or texture or the characteristic based on semantic interpretation.
2. Every region should be connected and non- overlapping i.e. any two pixels in a particular region should be connected by a line that does not leave the region.
3. It should not be possible to merge two adjacent regions to form a single homogeneous region.

The tasks such as object recognition & detection, feature extraction and classification are dependent on the quality of segmentation process. "Colour images can increase the quality of segmentation but complexity of the problem is also increased". Segmentation of a coloured image having different kinds of texture regions is a hard problem, particularly if an exact texture field is to be computed and a decision is to be made regarding optimum number of segments in the image. The problem becomes further complicated if the image contains similar and/or "non-stationary texture fields".[R. Chellappa, C.L. Wilson]

Each pixel of coloured images is denoted by three component values i.e. Red, Green and Blue and as such these are more complex as far as segmentation is concerned, than grey scale images which have a single intensity value for a pixel. Coloured image segmentation can solve many contemporary problems in medical imaging, mining and mineral imaging, bioinformatics, and material sciences with the increasing processing speed and decreasing computational cost, processing of colour images has drawn attention of many researchers. "Colour

images have much more information than grey scale images. Each pixel in a colour image has information about brightness, hue and saturation. Many models such as, RGB (red, green, blue), CMY (cyan, magenta, yellow), HSV (hue, saturation, and intensity), YIQ, HSI and many others represent the colours. Several colour spaces have been used for image segmentation and no general advantage of one colour space has yet been found. Many of the colour image segmentation algorithms are derived from methods of grey-scale image segmentation. However, colour creates a more complete representation of an image and exploiting this fact can result in a more reliable segmentation".

Segmentation in colour images is easy when all the objects are having distinct colours and boundaries but becomes a complex problem in following cases: (i) When images have "many complex objects with less distinct colours" (ii) Gradual variations are found in colour, illumination, shading and textures Colour images have regions with different kinds of textures and as such have following problems in their segmentation [120]: (i) Finding "optimum number of segmentation areas in an image when it contains similar and/or non-stationary texture fields". (ii) Computing exact texture fields. [R. Chellappa, C.L. Wilson]

3. PROPOSED SYSTEM

3.1 Python:

Python is a high-level, interpreted scripting language developed in the late 1980s by Guido van Rossum at the National Research Institute for Mathematics and Computer Science in the Netherlands. The initial version was published at the alt sources newsgroup in 1991, and version 1.0 was released in 1994.

Python is Interpreted

This makes for a quicker development cycle because you just type in your code and run it, without the intermediate compilation step.

One potential downside to interpreted languages is execution speed. Programs that are compiled into the native language of the computer processor tend to run more quickly than interpreted programs. For some applications that are particularly computationally intensive, like graphics processing or intense number crunching, this can be limiting.

In practice, however, for most programs, the difference in execution speed is measured in milliseconds, or seconds at most, and not appreciably noticeable to a human user. The expediency of coding in an interpreted language is typically worth

Python is Free

The Python interpreter is developed under an OSI-approved open-source license, making it free to install, use, and distribute, even for commercial purposes.

A version of the interpreter is available for virtually any platform there is, including all flavors of Unix, Windows, macOS, smartphones and tablets, and probably anything else you ever heard of. A version even exists for the half dozen people remaining who use OS/2.

Python is Portable

Because Python code is interpreted and not compiled into native machine instructions, code written for one platform will work on any other

platform that has the Python interpreter installed. (This is true of any interpreted language, not just Python.)

Installation of python

The Python download requires about 25 Mb of disk space; keep it on your machine, in case you need to re-install Python. When installed, Python requires about an additional 90 Mb of disk space

1. Download Python installer from this link
   https://www.python.org/downloads/
   The following page will appear in your browser.

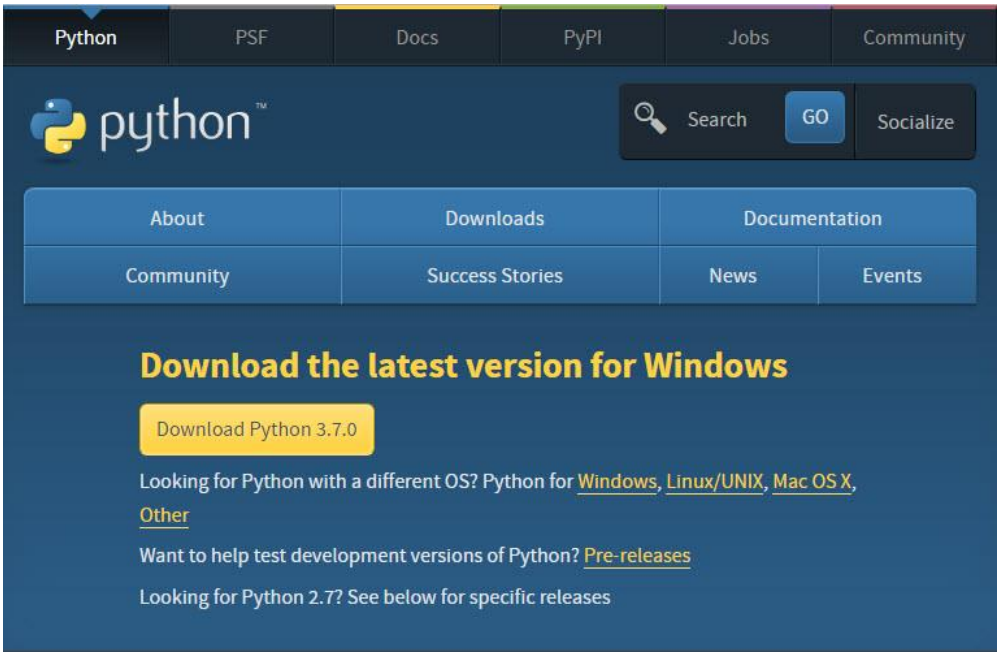
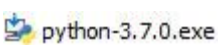


Fig 3.1

1. Click the **Download Python 3.7.0** button.

   The file named **python-3.7.0.exe** should start downloading into your standard download folder. This file is about 30 Mb so it might take a while to download fully if you are on a slow internet connection (it took me about 10 seconds over a cable modem).

   The file should appear as



2. Move this file to a more permanent location, so that you can install Python (and reinstall it easily later, if necessary).
3. Feel free to explore this webpage further; if you want to just continue the installation, you can terminate the tab browsing this webpage.
4. Start the **Installing** instructions directly below.

Installing Python:

1) Double-click the icon labeling the file **python-3.7.0.exe**.

An **Open File - Security Warning** pop-up window will appear.

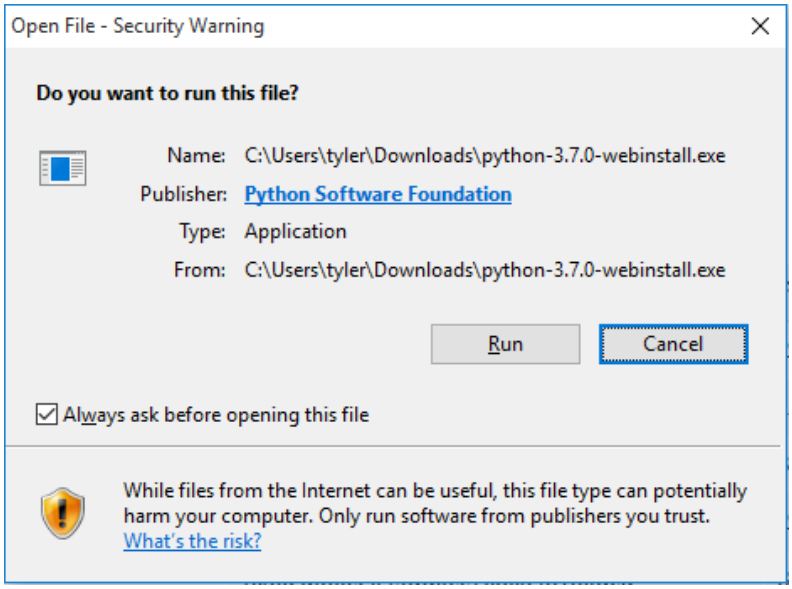


Fig 3.2

2) Click **Run**.

A **Python 3.7.0 (32-bit) Setup** pop-up window will appear.

Fig 3.3

1. Ensure that the **Install launcher for all users (recommended)** and the **Add Python 3.7 to PATH** checkboxes at the bottom are checked.

   If the Python Installer finds an earlier version of Python installed on your computer, the **Install Now** message may instead appear as **Upgrade Now** (and the checkboxes will not appear).

2. Highlight the **Install Now** (or **Upgrade Now**) message, and then click it.

   A **User Account Control** pop-up window will appear, posing the question **Do you want the allow the following program to make changes to this computer?**

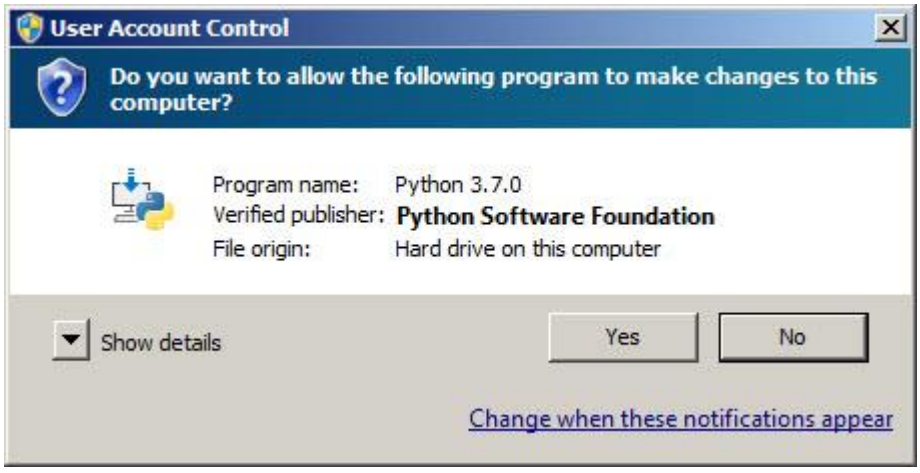


Fig 3.4

3. Click the **Yes** button.

   A new **Python 3.7.0 (32-bit) Setup** pop-up window will appear with a **Setup Progress** message and a progress bar.

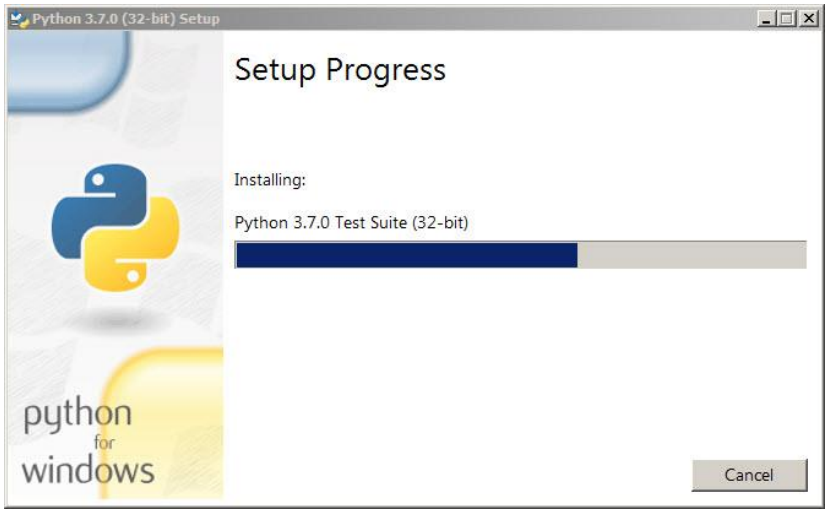


Fig3.5

During installation, it will show the various components it is installing and move the progress bar towards completion. Soon, a new **Python 3.7.0 (32-bit) Setup** pop-up window will appear with a **Setup was successfully** message.

Fig 3.6

4. Click the **Close** button.

Python should now be installed.

Verifying

To try to verify installation,

1. Navigate to the directory **C:\Users\Pattis\AppData\Local\Programs\Python\Python37-32** (or to whatever directory Python was installed: see the pop-up window for Installing step 3).
2. Double-click the icon/file **python.exe**.

The following pop-up window will appear.

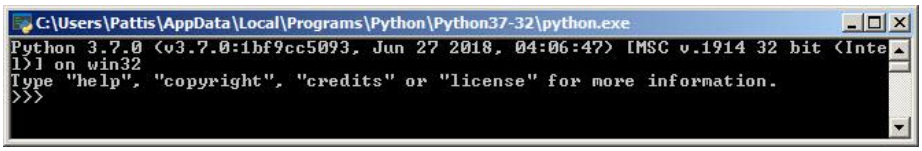


Fig 3.7

A pop-up window with the title **C:\Users\Pattis\AppData\Local\Programs\Python\Python37-32** appears, and inside the window; on the first line is the text **Python 3.7.0 ...** (notice that it should also say 32 bit). Inside the window, at the bottom left, is the prompt >>>: type **exit()** to this prompt and press **enter** to terminate Python.

Installing Python Modules

Key terms

- pip is the preferred installer program. Starting with Python 3.4, it is included by default with the Python binary installers.
- A *virtual environment* is a semi-isolated Python environment that allows packages to be installed for use by a particular application, rather than being installed system wide.
- venv is the standard tool for creating virtual environments, and has been part of Python since Python 3.3. Starting with Python 3.4, it defaults to installing pip into all created virtual environments.
- Virtualenv is a third party alternative (and predecessor) to venv. It allows virtual environments to be used on versions of Python prior to 3.4, which either don't provide venv at all, or aren't able to automatically install pip into created environments.
- The Python Packaging Index is a public repository of open source licensed packages made available for use by other Python users.
- The Python Packaging Authority is the group of developers and documentation authors responsible for the maintenance and evolution of the standard packaging tools and the associated metadata and file format standards. They maintain a variety of tools, documentation, and issue trackers on both GitHub and Bit Bucket.
- Disputes are the original build and distribution system first added to the Python standard library in 1998. While direct use of disputes is being phased out, it still

laid the foundation for the current packaging and distribution infrastructure, and it not only remains part of the standard library, but its name lives on in other ways (such as the name of the mailing list used to coordinate Python packaging standards development).

3.2 Basic Usage

The standard packaging tools are all designed to be used from the command line.

The following command will install the latest version of a module and its dependencies from the Python Packaging Index:

```
python -m pip install SomePackage
```

It's also possible to specify an exact or minimum version directly on the command line. When using comparator operators such as >, < or some other special character which get interpreted by shell, the package name and the version should be enclosed within double quotes:

```
python -m pip install SomePackage==1.0.4    # specific version
python -m pip install "SomePackage>=1.0.4"  # minimum version
```

Normally, if a suitable module is already installed, attempting to install it again will have no effect. Upgrading existing modules must be requested explicitly:

```
python -m pip install --upgrade SomePackage
```

3.3 Open CV :

Open CV (Open Source Computer Vision Library) is released under a BSD license and hence it's free for both academic and commercial use. It has C++, Python and Java interfaces and supports Windows, Linux, Mac OS, iOS and Android. OpenCV was designed for computational efficiency and with a strong focus on real-time applications. Written in optimized C/C++, the library can take advantage of multi-core processing. Enabled with Open CL, it can take advantage of the hardware acceleration of the underlying heterogeneous compute platform.

Installation of Open-CV

```
Python –m pip install opencv-python
```

DIGITAL IMAGE

A digital image is a numerical 2D or 3D representation of a real physical object or scene, from which we can obtain an accurate spatial (geometric) and/or spectral (for the case of a hyper spectral image) representation with sufficient detail (resolution) for processing, compression, storage, printing and display. A digital image may be of vector or raster type, depending on whether or not the image resolution is fixed.

Vector graphics formats are complementary to raster graphics, which are the representation of images based on mathematical expressions, as are typically used in computer graphics for images made up of vectors (arrows of direction, points, and lines) that define shapes, as compared to the individual pixels used to represent a raster image. A vector image is resolution independent, which means the image can be enlarged or shrunken without affecting the output quality. The term digital images usually refer to raster images, which are also called bitmap images. Moreover, it is also applied to data associated to points scattered over a three dimensional region, such as produced by tomographic equipment. In that case, each datum is called a voxel. The most commonly used method for the creation of raster images (i.e., digital imaging or image acquisition) is digital photography with a CCD camera in a process called digitization.

The digitization process requires the mapping of the image on a grid and a quantization of the intensity level. Digitization is the process of converting an analog signal into a digital signal, known as an A/D (analog to digital) conversion. For raster images, an analog voltage signal (from any of several types of imaging sensors), proportional to the amount of light reflected or transmitted by an item being digitized, is divided into discrete numeric values (Cattleman 1979). In other words, the process converts an image into a series of small picture square elements or pixels that are either black or white (binary), a specific shade of gray (grayscale) or color. Each pixel is represented by a single or series of binary digits, either 1 s or 0 s. By measuring the color (or gray-level for black and white photos) of an image at a large number of points, we can create a digital approximation of the image from which a copy of the original can be reconstructed.

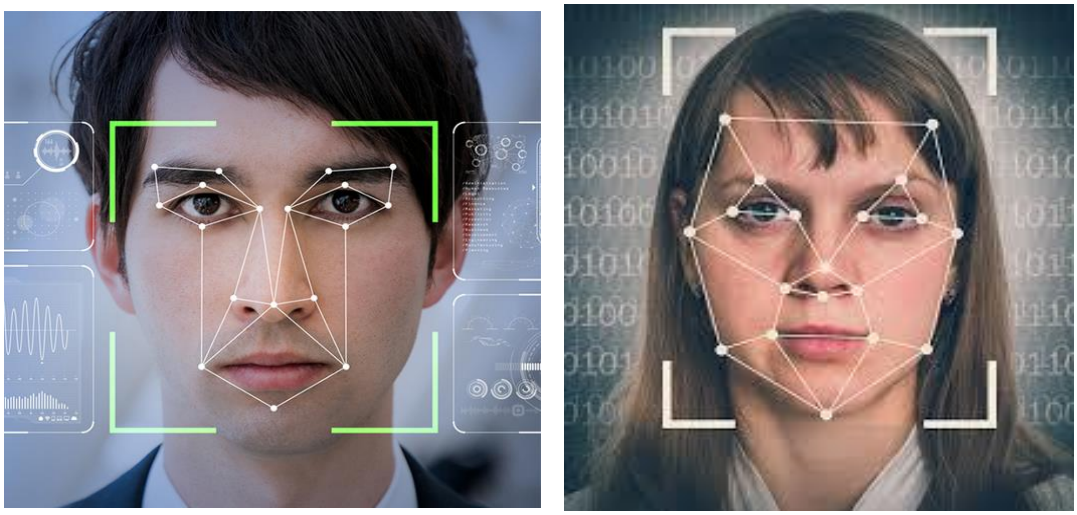


Fig 3.8

Fig. 1 depicts an example of the process for creation of a digital image.

Raster images are electronic files that consist of discrete picture elements, called pixels (short for picture elements). Associated with each pixel is a number that is the average radiance (or brightness) of a relatively small area within a scene, representing the color or gray-level at a single point in the image.

Digital image is how computer record the photos and pictures. The smallest unit is called pixel, normally consists of a 0-255 value for gray images(0:black, 255 white). For color images, each pixel has three 0-255 values, representing RBG. So the whole picture or photo in computer is actually a data table, or matrix, with different size of pixels, e.g. 1920*1080. [A. Lanitis, C. J. Taylor]

Digital images are made of picture elements called pixels. Typically, pixels are organized in an ordered rectangular array. The size of an image is determined by the dimensions of this pixel array. The image width is the number of columns, and the image height is the number of rows in the array. Thus the pixel array is a matrix of M columns x N rows. To refer to a specific pixel within the image matrix, we define its coordinate at x and y. The coordinate system of image matrices defines x as increasing from left to right and y as increasing from top to bottom. A digital image is a numeric representation of a two-dimensional image. Depending on whether the image resolution is fixed, it may be of vector or raster type. By itself, the term "digital image" usually refers to raster images or bitmapped images.

4. SYSTEM ANALYSIS AND DESIGN

There have been many attempts to solve human face detection problem. The early approaches are aimed for gray level images only, and image pyramid schemes are necessary to scale with unknown face sizes. View-based detectors are popular in this category, including Rowley's neural networks classifier, Sung and Poggio's correlation templates matching scheme based on image invariants and Eigen-face decomposition. Model based detection is another category of face detectors.

4.1 Need To Conversion Of Colour Image Into Grayscale

In digital photography, computer-generated imagery, and color metric, a **grayscale** or **greyscale** image is one in which the value of each pixel is a single sample representing only an *amount* of light, that is, it carries only intensity information. Grayscale images, a kind of black-and-white or **gray monochrome**, are composed exclusively of shades of gray. The contrast ranges from black at the weakest intensity to white at the strongest.

Grayscale images are distinct from one-bit bi-tonal black-and-white images which, in the context of computer imaging, are images with only two colors: black and white (also called *bi-level* or *binary images*). Grayscale images have many shades of gray in between.

While digital images can be saved as grayscale (or black and white) images, even color images contain grayscale information. This is because each pixel has a luminance value, regardless of its color. Luminance can also be described as brightness or intensity, which can be measured on a scale from black (zero intensity) to white (full intensity). Most image file formats support a minimum of 8-bit grayscale, which provides 2^8 or 256 levels of luminance per pixel. Some formats support 16-bit grayscale, which provides 2^16 or 65,536 levels of luminance.

Following are the various reasons for gray scale conversion: -

1. Signal to noise. For many applications of image processing, color information doesn't help us identify important edges or other features. There are exceptions. If there is an edge (a step change in pixel value) in hue that is hard to detect in a grayscale image, or if we need to identify objects of known hue (orange fruit in front of green leaves), then color information could be useful. If we don't need color, then we can consider it

noise. At first it's a bit counterintuitive to "think" in grayscale, but you get used to it.

2. Complexity of the code. If you want to find edges based on luminance AND chrominance, you've got more work ahead of you. That additional work (and additional debugging, additional pain in supporting the software, etc.) is hard to justify if the additional color information isn't helpful for applications of interest.

3. For learning image processing, it's better to understand grayscale processing first and understand how it applies to multichannel processing rather than starting with full color imaging and missing all the important insights that can (and should) be learned from single channel processing.

4. Difficulty of visualization. In grayscale images, the watershed algorithm is fairly easy to conceptualize because we can think of the two spatial dimensions and one brightness dimension as a 3D image with hills, valleys, catchment basins, ridges, etc. "Peak brightness" is just a mountain peak in our 3D visualization of the gray scale image. There are a number of algorithms for which an intuitive "physical" interpretation helps us think through a problem. In RGB, HSI, Lab, and other color spaces this sort of visualization is much harder since there are additional dimensions that the standard human brain can't visualize easily. Sure, we can think of "peak redness," but what does that mountain peak look like in an (x,y,h,s,i) space? Ouch. One workaround is to think of each color variable as an intensity image, but that leads us right back to gray scale image processing.

5. Color is complex. Humans perceive color and identify color with deceptive ease. If you get into the business of attempting to distinguish colors from one another, then you'll either want to (a) follow tradition and control the lighting, camera color calibration, and other factors to ensure the best results, or (b) settle down for a career-long journey into a topic that gets deeper the more you look at it, or (c) wish you could be back working with gray scale because at least then the problems seem solvable.[A. Lanitis, C. J. Taylor]

6. Speed. With modern computers, and with parallel programming, it's possible to perform simple pixel-by-pixel processing of a megapixel image in milliseconds. Facial recognition, OCR, content-aware resizing, mean shift segmentation, and other tasks can take much longer than that. Whatever processing time is required to manipulate the image or squeeze some useful data from it, most customers/users want it to go faster. If we make the hand-wavy assumption that processing a three-channel color image takes three times as long as processing a gray scale image--or maybe four times as long, since we may create a separate luminance channel--then that's not a big deal if we're processing video images on the fly and each frame can be processed in less than 1/30th or 1/25th of a second. But if we're analyzing thousands of images from a database, it's great if we can save ourselves processing time by resizing images, analyzing only portions of images, and/or eliminating color channels we don't need. Cutting processing time by a factor of three to four can mean the difference between running an 8-hour overnight test that ends before you get back to work, and having your computer's processors pegged for 24 hours straight.[A. Lanitis, C. J. Taylor]

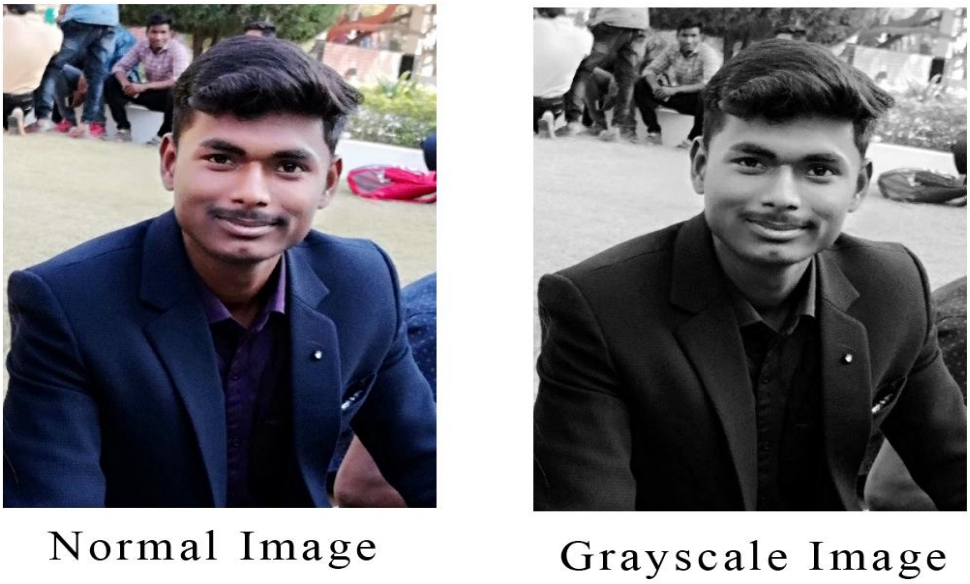


Fig 4.1

Fig. Images saved at different dpi levels while keeping the same size.

(a) typical effects on a color image;

(b) conversion of color image into grayscale image.

Images saved with the lowest dpi levels appear blurred and have a reduced contrast. By decreasing the image resolution of the apple tissue (Fig. 2 b) the finest details of the tissue microstructure are lost, making it more difficult for the visualization and analytical determination of the smallest structures.

4. 2 Box formation

Around each white area a box is formed using 8-connectivity as shown in figure

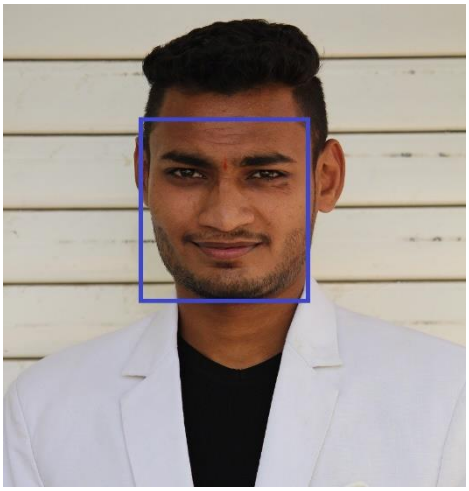


Fig 4.2

4.3 Unwanted box Rejection:-

Many non-face areas are also selected like hands and areas having colours similar to the skin colour. So those boxes need to be rejected. 2 approaches are used for this purpose as mentioned below.

4.3.1 Thresholding

As faces generally don't fall far below the image, lower boxes around 100 pixels below the image were discarded. mage thresholding is a simple, yet effective, way of partitioning an image into a foreground and background. This image analysis technique is a type of image segmentation that isolates objects by converting grayscale images into binary images. Image thresholding is most effective in images with high levels of contrast.

Fig 4.3

4.3.2 Image matching

The Eigen faces of a set of images are obtained and the mean Eigen face is taken as the reference for a face structure. All the images in the box areas are compared with the Eigen face and the correlation between them is found out. Non-face areas will have low correlation while face areas will have high correlation. Then the boxes having less value of correlation are discarded. Since the boxes can be of any size, the Eigen face is stored in different sizes starting from 30 pixels to 220 pixels at the step of 10 pixels (boxes are square boxes). The test images for the Eigen face generation are taken as 100x100 square images.

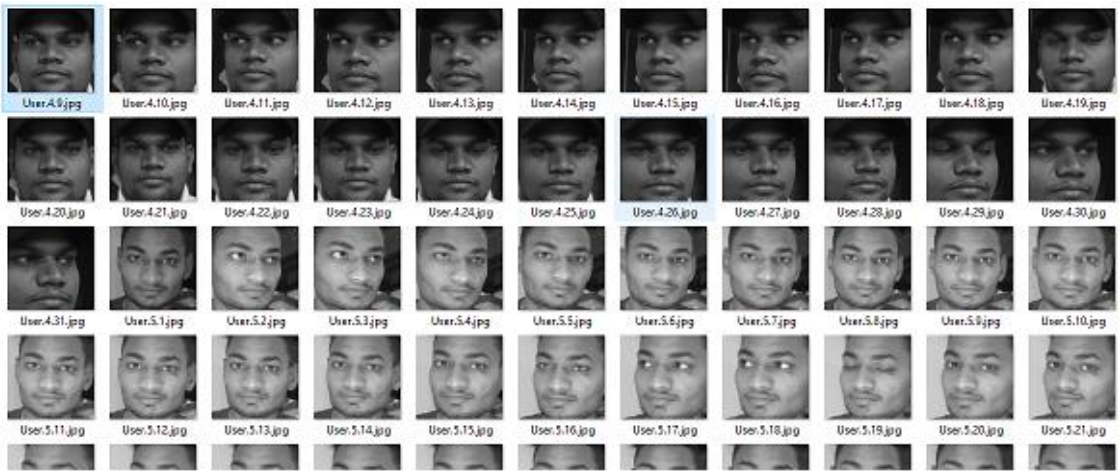


Fig 4.4 :- Different Eigen face

Data Flow Diagram:

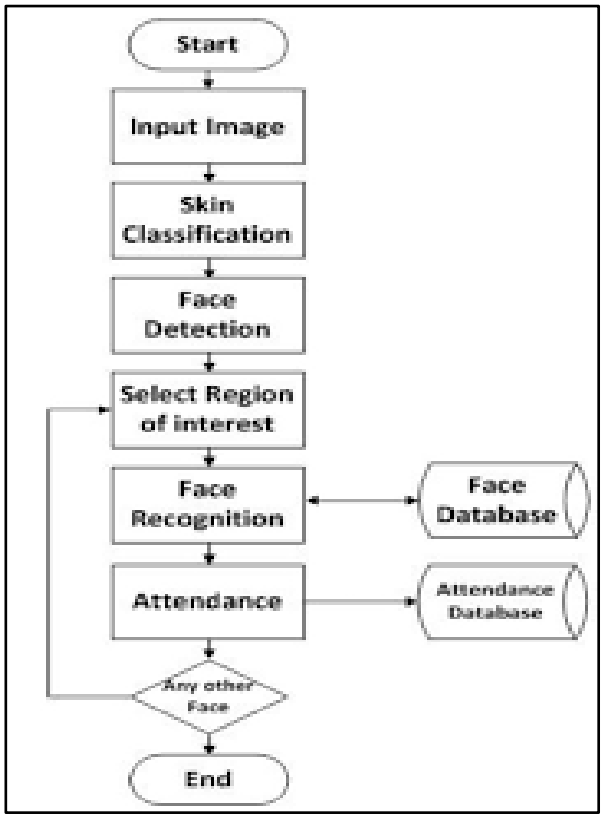


Fig 4.5 DFD Of Project

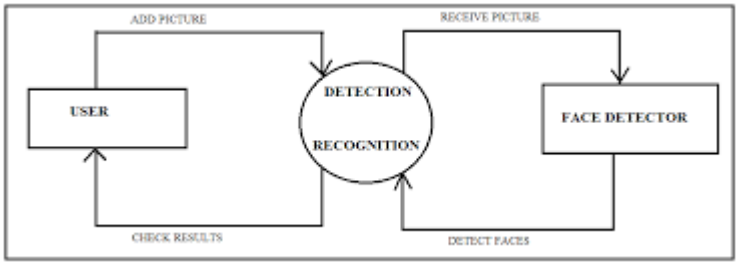


Fig 4.6 DFD For Detecting face in Database

5. PROJECT OBJECTIVES

- **Visualization:** Make the invisible parts of the objects visible.
- **Image sharpening and restoration:** Increase the quality of the image
- **Image retrieval:** Search and find out the desired images from the large database
- **Measurement of the pattern:** Measure every parts or object of an image
- **Image Recognition:** Recognize the components of the image

6. SYSTEM IMPLEMENTATION

6.1 Image Analytics for Classifying Data Sets for large Class

We Create the project on Image Analytics for Classifying Data Sets for large Class in Face Detection and Recognition. In this project we creating a software in which it detects the face and recognize using the face detection algorithm. This reduce the difficulty in identifying face of the person used. The face recognition is done using the feature base cascade classifiers using Eigen face algorithm. In addition to the face recognition this project also enhances the process by providing actual output.

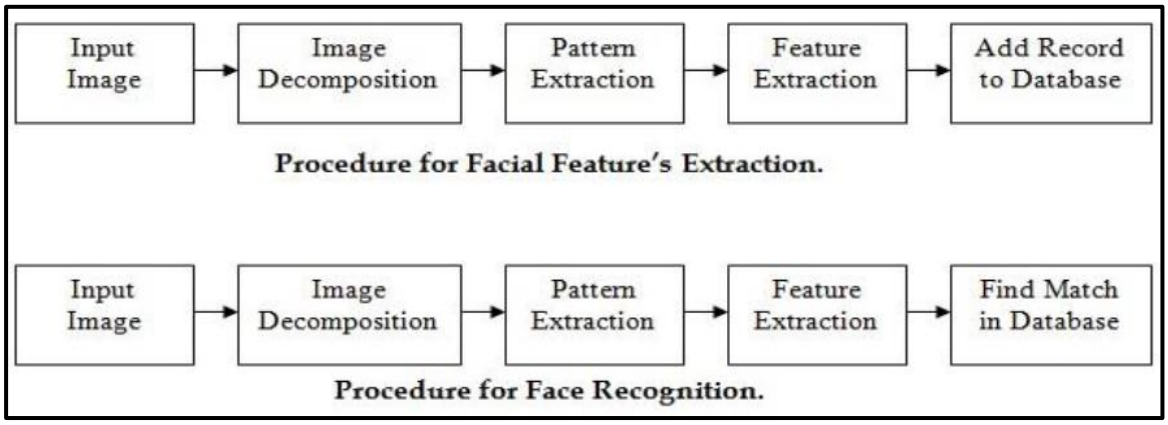


Fig 6.1

Firstly, in facial feature detection is detecting the face. This requires analyzing the entire image. The second step is using the isolated face to detect each feature. Since each portion of the image used to detect a feature is much smaller than that of the whole image, detection of all three facial features takes less time on average than detecting the face itself. Using a raspberry pi B+ processor to analyze a 320 by 240 image, a frame rate of 3 frames per second was achieved. [T.J. Sejnowski]

With the detection of facial features, the next goal is to research the ability for more precise details, like some individual points, of the facial features to be gathered. Those will be use to differentiate general human emotions, like happiness and sadness and other emotions. Recognition of human emotions would require accurate detection and analysis of the various elements of a human

face, like the brow and the mouth, to determine an individual's current expression.

The expression can then be compared to what is considered to be the basic signs of an emotion in all human beings. This research will be used in the field human-computer interaction to analyze the emotions one exhibits while interacting with a user interface which was not yet experimented before in the world of science and advancement Since a frame rate of 5 frames per second was achieved in facial detection only by using a much faster processor, regionalization provides a tremendous increase in efficiency in facial feature detection. Regionalization also greatly increased the accuracy of the detection. All false positives were eliminated, giving a detection rate of around 95% for the eyes and nose. The mouth detection has a lower rate due to the minimum size required for detection

Color is complex. Humans perceive color and identify color with deceptive ease. If you get into the business of attempting to distinguish colors from one another, then you'll either want to (a) follow tradition and control the lighting, camera color calibration, and other factors to ensure the best results, or (b) settle down for a career-long journey into a topic that gets deeper the more you look at it, or (c) wish you could be back working with gray scale because at least then the problems seem solvable.

Screenshots:

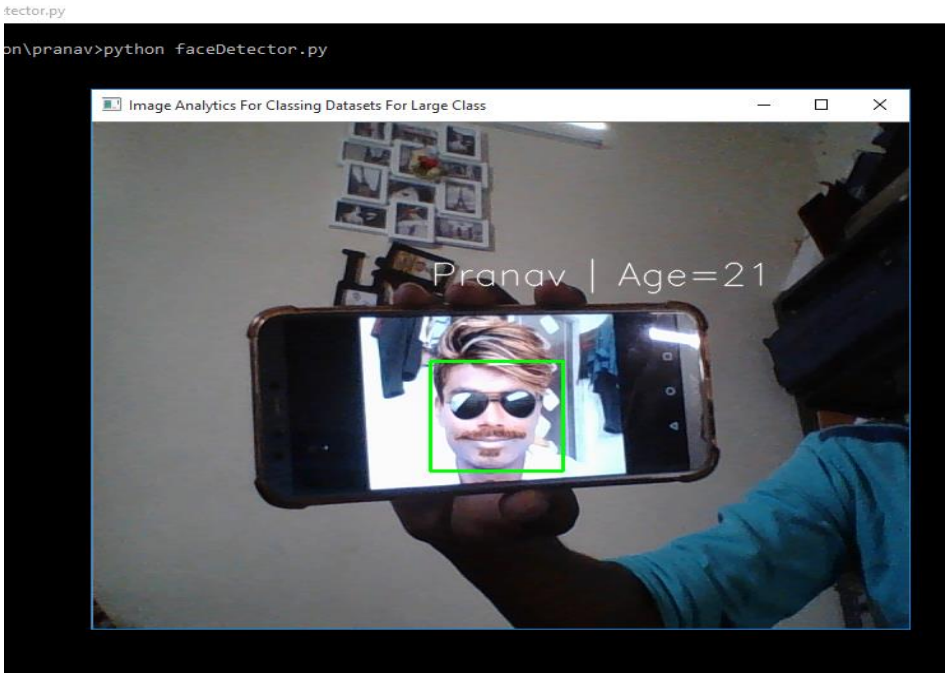


Fig 6.2

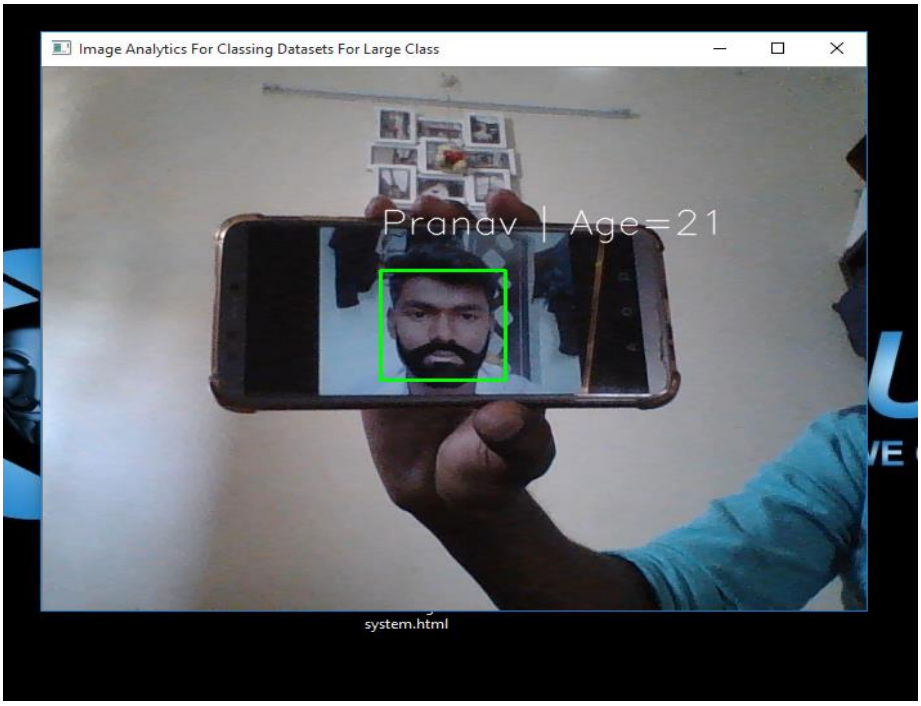


Fig 6.3

7. HARDWARE AND SOFTWARE REQUIRMENTS

Hardware Requirements:

1) Intel Core windows 7/8/10.
2) 1 GB Ram
3) Graphics Card or Integrated Graphics upto 512 Mb.

Software Requirements :

1) Pyton 3.6
2) OpenCV
3) Pillow
4) Pycharm

8. ADNTAGES & DISADVANTAGES

1. Advantages

2. Digital images can be processed by digital computers.
3. Images can be given more sharpness and better visual appearance
4. Minor errors can be rectified.
5. remove noises.
6. Image can be made available in any desired formats like black and white, negative image.

2. Disadvantages

1. It's very costly depending on the system used, the number of detectors purchased.

9. APPLICATIONS

1. The applications of digital image analysis are continuously expanding through all areas of science and industry including:
2. Assay micro plate reading, such as detecting where a chemical was manufactured.
3. Astronomy, such as calculating the size of a planet.
4. Defense
5. Error level analysis
6. Filtering
7. Machine vision, such as to automatically count items in a factory conveyor belt.
8. Materials science, such as determining if a metal weld has cracks.
9. Medicine, such as detecting cancer in a mammography scan.
10. Metallography, such as determining the mineral content of a rock sample.
11. Microscopy, such as counting the germs in a swab.
12. Optical character recognition, such as automatic license plate detection.
13. Remote sensing, such as detecting intruders in a house, and producing land cover/land use map.
14. Robotics, such as to avoid steering into an obstacle.
15. Security, such as detecting a person's eye color or hair color

10.CONCLUSION

The computational models, which were implemented in this project, were chosen after extensive research, and the successful testing results confirm that the choices made by the researcher were reliable. The system with manual face detection and automatic face recognition did not have a recognition accuracy over 90%, due to the limited number of Eigen faces that were used for the PCA transform. This system was tested under very robust conditions in this experimental study and it is envisaged that real-world performance will be far more accurate. The fully automated frontal view face detection system displayed virtually perfect accuracy and in the researcher's opinion further work need not be conducted in this area.

The fully automated face detection and recognition system was not robust enough to achieve a high recognition accuracy. The only reason for this was the face recognition subsystem did not display even a slight degree of invariance to scale, rotation or shift errors of the segmented face image. This was one of the system requirements identified in section 2.3. However, if some sort of further processing, such as an eye detection technique, was implemented to further normalise the segmented face image, performance will increase to levels comparable to the manual face detection and recognition system. Implementing an eye detection technique would be a minor extension to the implemented system and would not require a great deal of additional research. All other implemented systems displayed commendable results and reflect well on the deformable template and Principal Component Analysis strategies.

11. SCOPE FOR FUTURE WORK

The use of spherical canonical images allows us to perform matching in the spherical harmonic transform domain, which does not require preliminary alignment of the images. The errors introduced by embedding into an expressional space with some predefined geometry are avoided. In this facial expression recognition setup, end-to-end processing comprises the face surface acquisition and reconstruction, smoothening, sub sampling to approximately 2500 points. Facial surface cropping measurement of large positions of distances between all the points using a parallelized parametric version is utilized.

The general experimental evaluation of the face expressional system guarantees better face recognition rates. Having examined techniques to cope with expression variation, in future it may be investigated in more depth about the face classification problem and optimal fusion of colour and depth information. Further study can be laid down in the direction of allele of gene matching to the geometric factors of the facial expressions. The genetic property evolution framework for facial expressional system can be studied to suit the requirement of different security models such as criminal detection, governmental confidential security breaches etc.

REFERENCES

1. http://etc.usf.edu/techease/win/images/what-is-bit-depth/
2. R. Chellappa, C.L. Wilson, and S. Sirohey, "Human and machine recognition of faces: A survey," Proc. IEEE, vol. 83, pp. 705–740, 1995.
3. H. Wechsler, P. Phillips, V. Bruce, F. Soulie, and T. Huang, Face Recognition: From Theory to Applications, Springer-Verlag, 1996.
4. W. Zhao, R. Chellappa, A. Rosenfeld, and P.J. Phillips, "Face recognition: A literature survey," CVL Technical Report, University of Maryland, 2000, ftp://ftp.cfar.umd.edu/TRs/CVLReports-2000/TR4167-zhao.ps.gz.
5. A. Lanitis, C. J. Taylor, and T. F. Cootes, "Towards automatic simulation of ageing effects on face images," IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 24, no. 4, pp. 442–455, 2002.
6. Yale University face database, http://cvc.yale.edu/projects/yalefaces/yalefaces.html.
7. M. Turk and A. Pentland, "Eigenfaces for recognition," Journal of Cognitive Neuroscience, vol. 3, no. 1, pp. 71–86, Mar. 1991.
8. M.S. Bartlett, H.M. Lades, and T.J. Sejnowski, "Independent component representations for face recognition," in Proceedings of SPIE, 1998, vol. 3299, pp. 528–539.