PROJECT REPORT ON

# AFFECTIVE VIDEO TAGGING

Under the guidance of **Prof. U.S. Tiwary**

**INDIAN INSTITUTE OF INFORMATION TECHNOLOGY**
**ALLAHABAD**

July – December, 2016

Submitted by:

Utkarsh Srivastava (IIT2014507)
Amit Vijay (IIT2014110)
Swapnaneel Nandy (IIT2014111)
Mohd. Abdullah (ISM2014004)
Shivam Beri (IIT2014159)

# CANDIDATES’ DELARATION

We hereby declare that the work presented in this project report entitled “**Affective Video Tagging**”, submitted end-semester report of 5th Semester project of B.Tech. (IT) at Indian Institute of Information Technology, Allahabad, is an authenticated record of our original work carried out from June 2016 to November 2016 under the guidance of

**Prof. U.S Tiwary**. Due acknowledgements have been made in the text to all other materials used. The project was done in full compliance with the requirements and constraints of the prescribed curriculum.

Place: Allahabad

Date: 28th Nov, 2016

Swapnaneel Nandy (IIT2014111)
Amit Vijay (IIT2014110)
Utkarsh Srivastava (IIT2014507)
Mohd. Abdullah (ISM2014004)
Shivam Beri (IIT2014159)

# CERTIFICATE

This is to certify that the above statement made by the candidates is correct to the best of my knowledge.

Date:<br>
Place: Allahabad

Prof. U.S Tiwary<br>
*Professor*<br>
IIIT-Allahabad

# Abstract

✓ These days, annotations play a vital role in the search and retrieval of contents in any content sharing system. Since the rate of increase of data on such system is exponential, it becomes challenging to retrieve contents that would be appealing based on emotional disposition of the users.

✓ Moreover, a system that efficiently retrieve multimedia contents according to the user's current mood, can be of great interest. There are several everyday applications of such systems. For example : enabling video delivery websites like YouTube and Netflix to show more convincing recommendations, better parental control by allowing parents to choose what their children watch by knowing the emotional content of the video.

✓ Affective video content analysis aims at automatic recognition of emotions elicited by videos. We aim to implement an efficient affective video analysis system through feature extraction and using those features to create a classification model using machine learning algorithms. The dataset used for training the system is LIRIS-ACCEDE, which has over 9800 videos.

# Table of Content

# 1. Introduction

As digital media world is evolving, services like social networks, search engines and internet-based multimedia repositories are getting more and more popular, and as a result, the multimedia content on the web is becoming larger and larger. Hence, developing optimal structures for efficient searching and retrieving in these multimedia repository becomes very significant. To assign tags to a given content, there are broadly two approaches namely explicit and implicit tagging. The first corresponds to a viewer's manual action of inputting keywords related to the data, and in the second method, viewers' don't insert tags but, automated analysis of the viewer's behavior is used to tag the video.

In most of the current generation social network-based systems, explicit tagging is used like YouTube. However, it's not the long term solution for tagging multimedia data because:

✓ As manual accurate glossary of the huge amount of multimedia content on web is obviously impractical, and therefore, a huge chunk of those content remains untagged, which hinders the search process and reduces the performance of search systems.

✓ Viewers' who tag the multimedia data don't usually target at bettering the efficiency of the present search systems. In fact, private and social catalysts often are the reason behind tagging: A self need-driven tag may be worthless to others.

✓ Some viewers' tag the content for miscellaneous purposes like spam tags for advertisement.

Hence, substitute plans have to be developed to overcome these drawbacks of explicit tagging. Implicit tagging is one of the most optimal solutions. A lot of research has been done on exploring signs which can be discovered by noticing and examining viewers' nature like attention, interest, and emotion. Such signs could then be used in automated tagging of multimedia content.

Across all the data that can be generated for automatic tagging, emotional detail of such content is considered optimal. It plays a paramount role for personalized content delivery. As an example, viewers' like to view video clips that have funny contents when they are depressed or sad to improve their mood. And some people will not want to watch video clips which will contain too much violence or frightening scenes.

To analyze a given video at an impactful level, some models for emotions have to be developed. Some dimensional models depict the parts of emotions and is generally illustrated in 2-D space with the emotions as coordinates. The two dimensions are: Valence (V) and Arousal (A). Valence is the measure of degree of attractiveness and it varies from negative to positive. Negative value will signify unattractiveness or unpleasant. Arousal gives information about the intensity of the emotion and ranges from excited to calm. We can use this dimensional model to develop a classification system to categorize videos clips based on their valence and arousal scale.

# 2. Literature Survey

| Sl. No. | Title | Author | Type | Year | Ideas |
|---|---|---|---|---|---|
| 1 | LIRIS-ACCEDE: A Video Database for Affective Content Analysis | YoannBaveye, Emmanuel Dellandrea, Christel Chamaret and Liming Chen. | Research | 2015 | Main paper of our database which describes the database and a brief description of features we should use |
| 2 | Affective Video Content Representation and Modeling | Alan Hanjalic, Li-Qun Xu. | Research | 2005 | Depiction of arousal - valence model and relevant features for video analysis |
| 3 | Prediction of the inter-observer visual congruency (IOVC) and application to image ranking | O. Le Meur, T. Baccino, and A. Roumy. | Research | 2011 | Explanation of depth of field and algorithms to calculate it |
| 4 | Photo and video quality evaluation: Focusing on the subject | Y. Luo and X. Tang | Research | 2008 | Algorithms for subject Extraction and for calculating compositional balance |
| 5 | Measuring colorfulness in natural images | David Haslera and Sabine Susstrunk | Research | 2003 | Algorithm to find the colorfulness metric |
| 6 | Relationship between color and emotion: a study of college students | Naz Kaya | Journal | 2004 | Relationship between color and emotion on valence and arousal scales |
| 7 | Two-Frame Motion Estimation Based on Polynomial Expansion | Gunnar Farneback | Research | 2003 | Gunnar Farneback algorithm for finding motion estimation and Global Activity |

| Sl. No. | Title | Author | Type | Year | Ideas |
|---|---|---|---|---|---|
| 8 | Polynomial Expansion for Orientation and Motion Estimation | Gunnar Farneback | Research | 2002 | Polynomial expansion representation of images for Gunner Farneback algorithm |
| 9 | Studying Aesthetics in Photographic Images Using a Computational Approach | RitendraDatta,Dhiraj Joshi,Jia Li and James Z. Wang | Research | 2006 | Algorithm for lighting metric |
| 10 | Color Harmonization for Videos | Nikhil Sawant and Niloy J. Mitra | Research | 2008 | Algorithm for color harmonization |
| 11 | Affect based indexing for multimedia data | Jones, Gareth J.F., and Ching Hua Chan | Research | 2006 | To study the scope of content based video retrieval |
| 12 | Affective Video Content Representation | Virginia Fernandez Arguedas | Research | 2005 | Effect of number of scene cuts per frame on arousal |

# 3. Valence-Arousal Model

The valence-arousal model or circumplex model of emotion was developed by James Russell. According to this model, emotion are distributed in a two-dimensional circular space. The two dimensions this circular space are Valence (V) and Arousal (A), here valence represents the horizontal axis and arousal represents the vertical axis. The center of the circle i.e. origin represent a medium level of arousal and a neutral valence.

**Valence**: Valence represents the positive or negative affectivity ororientation of the emotion. An example of a positive valence emotion is happy and that of a negative valence emotion is sadness.

**Arousal**: Arousal represents the intensity of the emotion. In other wordsit measures how calming or exciting the emotion is.

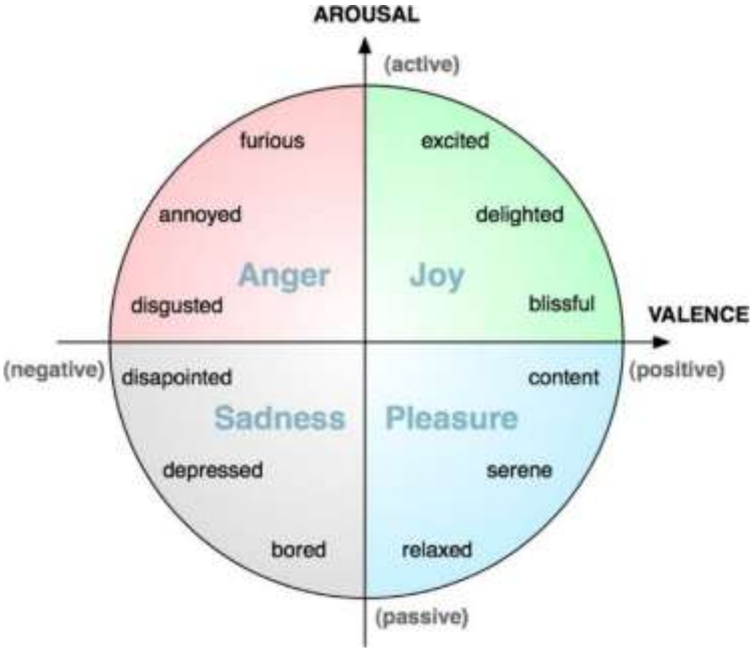


Figure 1: Valence-Arousal Model

# 4. FEATURE LIST

The features we have decided to extract for our model are:

- **Global Activity**: Measures apparent motion in the video.
- **Number of scene cuts per frame**: Measures shot cut rate of thevideo.
- **Zero Crossing Rate**: It is the rate of change of signs of signals.
- **Lighting**: Measures average luminescence of the video.
- **Colorfulness**: Measures average amount of color in the video.
- **Spectral Flatness**: It is the way to quantify how noisy or tonal a sound is.
- **Length of scene cuts** : Analyses the length of each scene in the video
- **Hue count**: Measures the unique number of hues in the video.
- **Depth of field**: is the measure of distance between closest andfarthest object in an image.
- **Compositional balance**: is measure of how well visual weight is balanced around the mid vertical axis.
- **Spectral Slope:** It is the measure of inclination of power spectrum.

# 5. EXTRACTED FEATURES

## 5.1 Compositional Geometry

Compositional balance is a crucial requirement for fine quality of photos. When different parts of an image get equal attention of viewer, we say that image is perfectly balanced in terms of composition.

The more balanced an image is the more pleasing it will be to the eyes. We measure the compositional balance in terms of a parameter called compositional geometry.

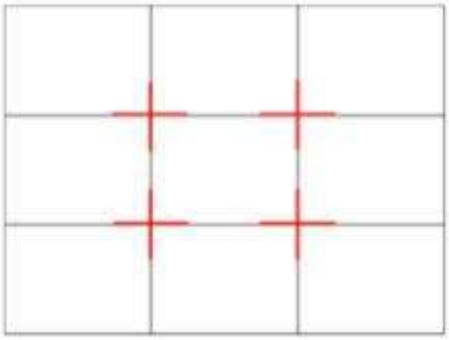
**Algorithm:**

We define a composition feature as

$$f_m = \min_{i=1,2,3,4}\{\sqrt{(C_{Rx} - P_{ix})^2/X^2 + (C_{Ry} - P_{iy})^2/Y^2}\},$$

$(C_{Rx}, C_{Ry})$ is the centroid coordinate of extracted binary subject region.

$(P_{ix}, P_{iu}), i = 1, 2, 3, 4,$ are four intersection points which we get by drawing 4 lines in the original image , and X is width of image and Y is height of the image respectively.



**SUBJECT EXTRACTION:**

Almost each photo has its lead character, called the subject, around which the image is created and this subject is the region of interest for viewer. Subject extraction of the given image is the first step of calculating **depth of field** and **compositional geometry.**

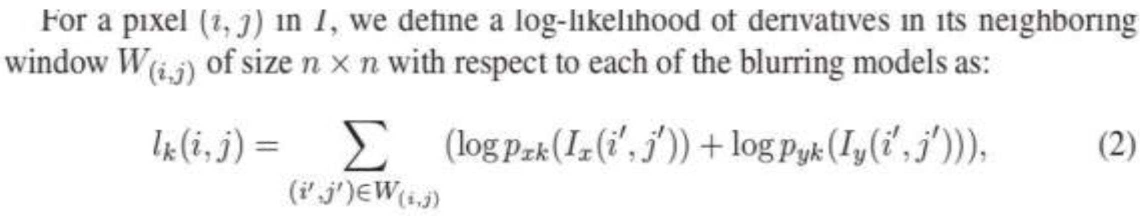
## Algorithm for Subject Extraction:

1. To blur the image firstly we choose a blurring kernel window [k x k] with blurring coeffs equal to 1/K*K.

2. Here $f_k$ denotes the blurring kernel window [k x k]. We convolve $f_k$ with I and to compute the vertical and horizontal derivatives from I ∗ $f_k$, we further convolve it with dx ([1,-1])and dy([{1},{-1}])respectively to get the vertical and horizontal derivatives:

$$p_{xk} \propto hist(I * f_k * d_x), \quad p_{yk} \propto hist(I * f_k * d_y)$$

3. Now we calculate the log-likelihood of the image as follows:



For a pixel $(i, j)$ in $I$, we define a log-likelihood of derivatives in its neighboring window $W_{(i,j)}$ of size $n \times n$ with respect to each of the blurring models as:

$$l_k(i,j) = \sum_{(i',j') \in W_{(i,j)}} (\log p_{xk}(I_x(i',j')) + \log p_{yk}(I_y(i',j'))), \qquad (2)$$

4. Next step is to find the blurring kernel ‘k’ that best suits the image:

$$k^*(i,j) = \arg\max_k l_k(i,j).$$

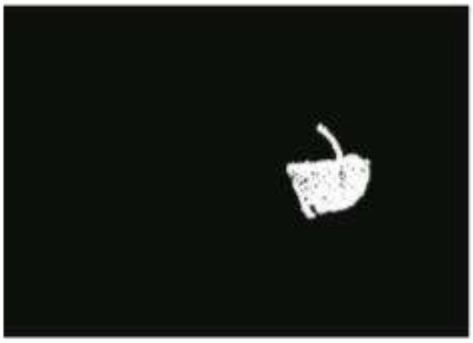
5. And then we create the binary image for every pixel (i,j) as follows:

$$U(i,j) = \begin{cases} 1, & k^*(i,j) = 1 \\ 0, & k^*(i,j) > 1. \end{cases}$$

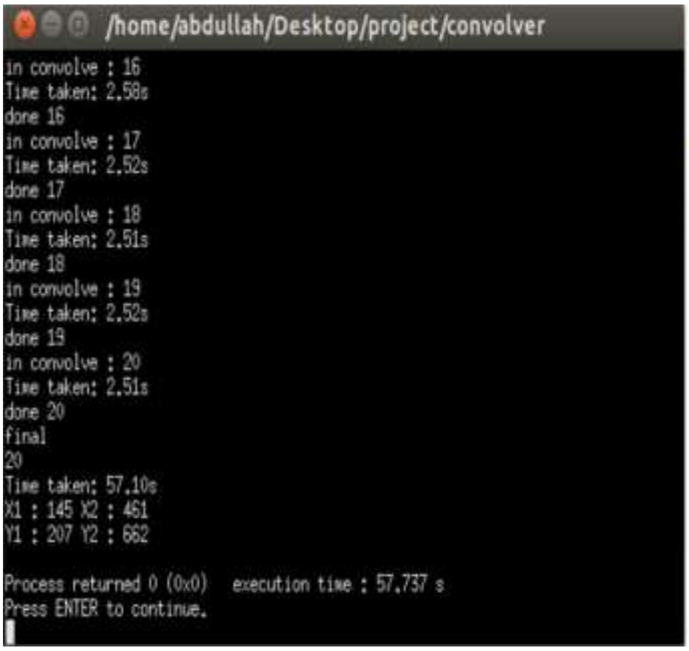
6. Do the summation over rows and column to get the energy projection for X-axis and Y-axis. Projection over X-axis is $U_y$ and projection over Y-axis is $U_x$.

$$U_x(i) = \sum_j U(i,j), \quad U_y(j) = \sum_i U(i,j).$$

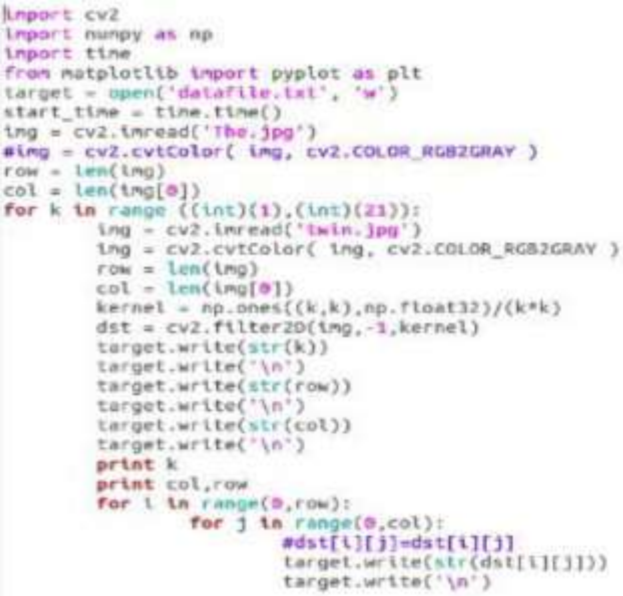
7. Now on X-axis we find two coordinates P and Q so as to get energy in portion [0,P] and [Q,N] equal and they should be equal to (1−c )/2 times overall energy E. Similar procedure we do to get Y-axis coordinate R and S. And then subject portion is rectangle [P+1, Q−1] × [R+1, S−1].



**SNIPPETS:**



```
in convolve : 16
Time taken: 2.58s
done 16
in convolve : 17
Time taken: 2.52s
done 17
in convolve : 18
Time taken: 2.51s
done 18
in convolve : 19
Time taken: 2.52s
done 19
in convolve : 20
Time taken: 2.51s
done 20
final
20
Time taken: 57.10s
X1 : 145 X2 : 461
Y1 : 207 Y2 : 662

Process returned 0 (0x0)   execution time : 57.737 s
Press ENTER to continue.
```



```
import cv2
import numpy as np
import time
from matplotlib import pyplot as plt
target = open('datafile.txt', 'w')
start_time = time.time()
img = cv2.imread('The.jpg')
#img = cv2.cvtColor( img, cv2.COLOR_RGB2GRAY )
row = len(img)
col = len(img[0])
for k in range ((int)(1),(int)(21)):
        img = cv2.imread('twin.jpg')
        img = cv2.cvtColor( img, cv2.COLOR_RGB2GRAY )
        row = len(img)
        col = len(img[0])
        kernel = np.ones((k,k),np.float32)/(k*k)
        dst = cv2.filter2D(img,-1,kernel)
        target.write(str(k))
        target.write('\n')
        target.write(str(row))
        target.write('\n')
        target.write(str(col))
        target.write('\n')
        print k
        print col,row
        for i in range(0,row):
                for j in range(0,col):
                        #dst[i][j]=dst[i][j]
                        target.write(str(dst[i][j]))
                        target.write('\n')
```

# 5.2 Colourfulness

The choice, contribution and contrast play a very important role in determining the affection conveyed by the video. The recent researches in psychology validate this fact, that color is an effective and precise non-verbal code of communication. Each color provokes specific subconscious reaction, thus evoking emotions. The Plutchik's wheel of emotion precisely describes the role played by color in eliciting emotions (Figure 2).

**For Instance**:

- Orange and Red are considered as "warm" colors, and they correspond to energetic and vigorous emotions such as 'anger' or 'fright'.
- Violet and Blue are "serene" colors, and they correspond to emotions related to 'comfort', 'insurance' and 'pleasure'.

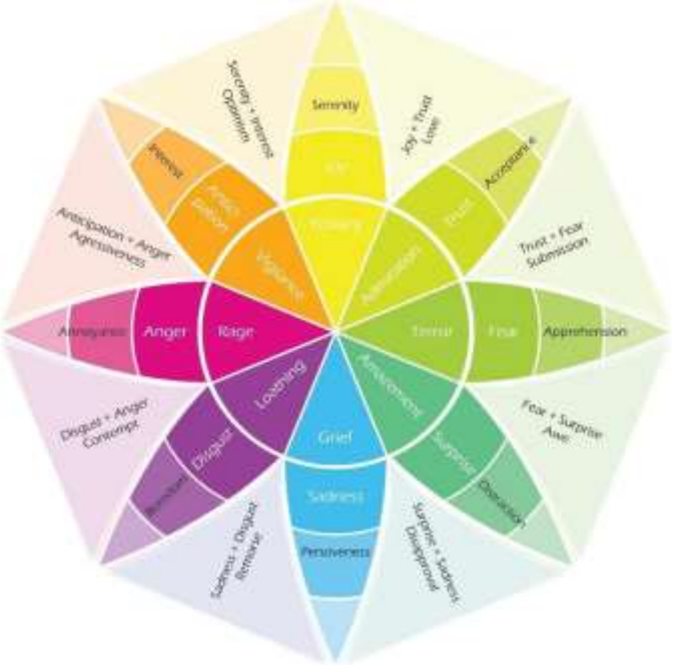
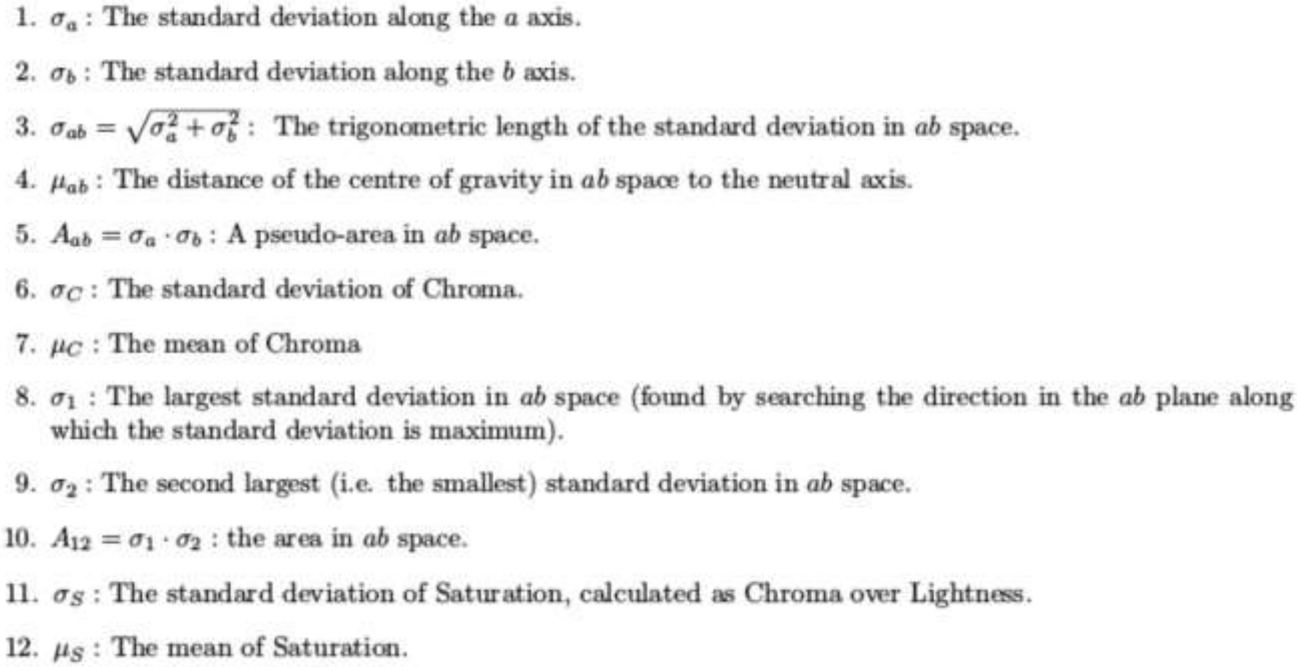


**Figure *2* : Plutchik's Wheel of Emotion**

## Computing the Colourfulness Metric:

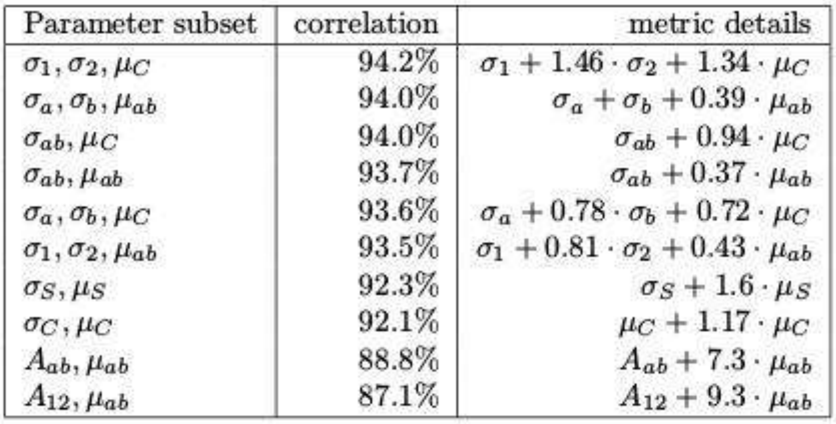
In the research paper "Measuring colorfulness of natural images", David Hasler and Sabine Susstrunk state few methods of calculation of the colorfulness metric. They propose that the metric can be calculated by a linear representation of a subset of the below quantities:



1. $\sigma_a$ : The standard deviation along the $a$ axis.
2. $\sigma_b$ : The standard deviation along the $b$ axis.
3. $\sigma_{ab} = \sqrt{\sigma_a^2 + \sigma_b^2}$ : The trigonometric length of the standard deviation in $ab$ space.
4. $\mu_{ab}$ : The distance of the centre of gravity in $ab$ space to the neutral axis.
5. $A_{ab} = \sigma_a \cdot \sigma_b$ : A pseudo-area in $ab$ space.
6. $\sigma_C$ : The standard deviation of Chroma.
7. $\mu_C$ : The mean of Chroma
8. $\sigma_1$ : The largest standard deviation in $ab$ space (found by searching the direction in the $ab$ plane along which the standard deviation is maximum).
9. $\sigma_2$ : The second largest (i.e. the smallest) standard deviation in $ab$ space.
10. $A_{12} = \sigma_1 \cdot \sigma_2$ : the area in $ab$ space.
11. $\sigma_S$ : The standard deviation of Saturation, calculated as Chroma over Lightness.
12. $\mu_S$ : The mean of Saturation.

## Result:

By choosing different subset of the above quantities, the result obtained are summarized in below table.



| Parameter subset | correlation | metric details |
|---|---:|---:|
| $\sigma_1, \sigma_2, \mu_C$ | 94.2% | $\sigma_1 + 1.46 \cdot \sigma_2 + 1.34 \cdot \mu_C$ |
| $\sigma_a, \sigma_b, \mu_{ab}$ | 94.0% | $\sigma_a + \sigma_b + 0.39 \cdot \mu_{ab}$ |
| $\sigma_{ab}, \mu_C$ | 94.0% | $\sigma_{ab} + 0.94 \cdot \mu_C$ |
| $\sigma_{ab}, \mu_{ab}$ | 93.7% | $\sigma_{ab} + 0.37 \cdot \mu_{ab}$ |
| $\sigma_a, \sigma_b, \mu_C$ | 93.6% | $\sigma_a + 0.78 \cdot \sigma_b + 0.72 \cdot \mu_C$ |
| $\sigma_1, \sigma_2, \mu_{ab}$ | 93.5% | $\sigma_1 + 0.81 \cdot \sigma_2 + 0.43 \cdot \mu_{ab}$ |
| $\sigma_S, \mu_S$ | 92.3% | $\sigma_S + 1.6 \cdot \mu_S$ |
| $\sigma_C, \mu_C$ | 92.1% | $\mu_C + 1.17 \cdot \mu_C$ |
| $A_{ab}, \mu_{ab}$ | 88.8% | $A_{ab} + 7.3 \cdot \mu_{ab}$ |
| $A_{12}, \mu_{ab}$ | 87.1% | $A_{12} + 9.3 \cdot \mu_{ab}$ |

A more easy and efficient algorithm for the calculation of colorfulness metric is stated below:

$$\begin{aligned}
rg &= R - G \\
yb &= \frac{1}{2}(R + G) - B \\
\hat{M}^{(3)} &= \sigma_{rgyb} + 0.3 \cdot \mu_{rgyb}, \\
\sigma_{rgyb} &:= \sqrt{\sigma_{rg}^2 + \sigma_{yb}^2}, \\
\mu_{rgyb} &:= \sqrt{\mu_{rg}^2 + \mu_{yb}^2},
\end{aligned}$$

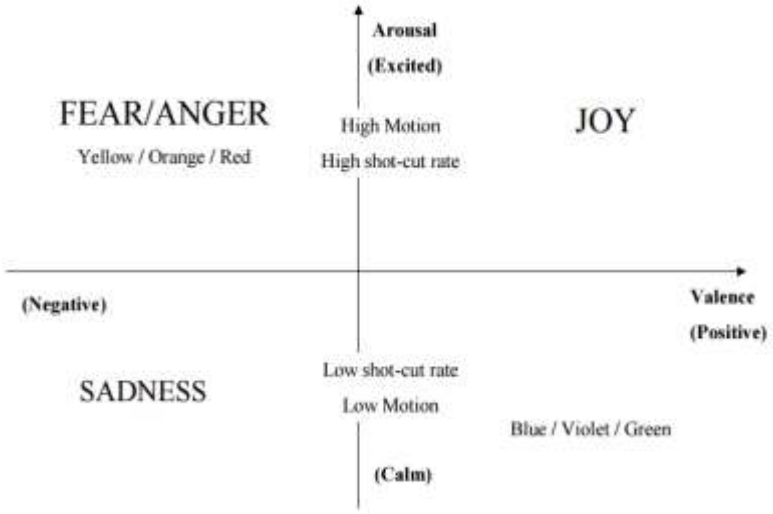
where 'σ' and 'µ' are the standard deviation and the mean value of the pixel cloud along the mentioned direction, respectively. Surprisingly, the correlation of M with the experimental data is equal to 95.3%, thus it represents a very efficient and easy method of calculating the metric.

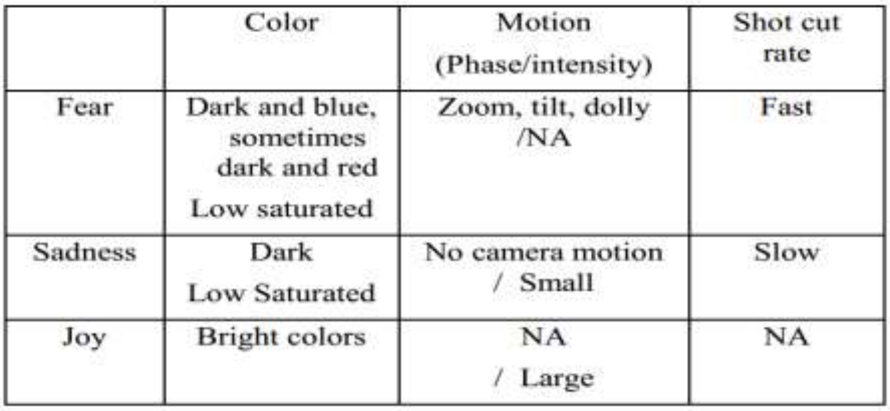
# 5.3 Number of scene cuts per frame

The number of scene cuts per frame can also be said as the shot cut rate of a video. It is one of the low level features for arousal in video indexing along with Color and Motion.

This is generally proportional to the arousal of the video. The more the scene cuts are in the video, the more will be the arousal of the video. In case if two video have same number of scene cuts, the one having less number of frames have more value for number of scene cuts per frame.

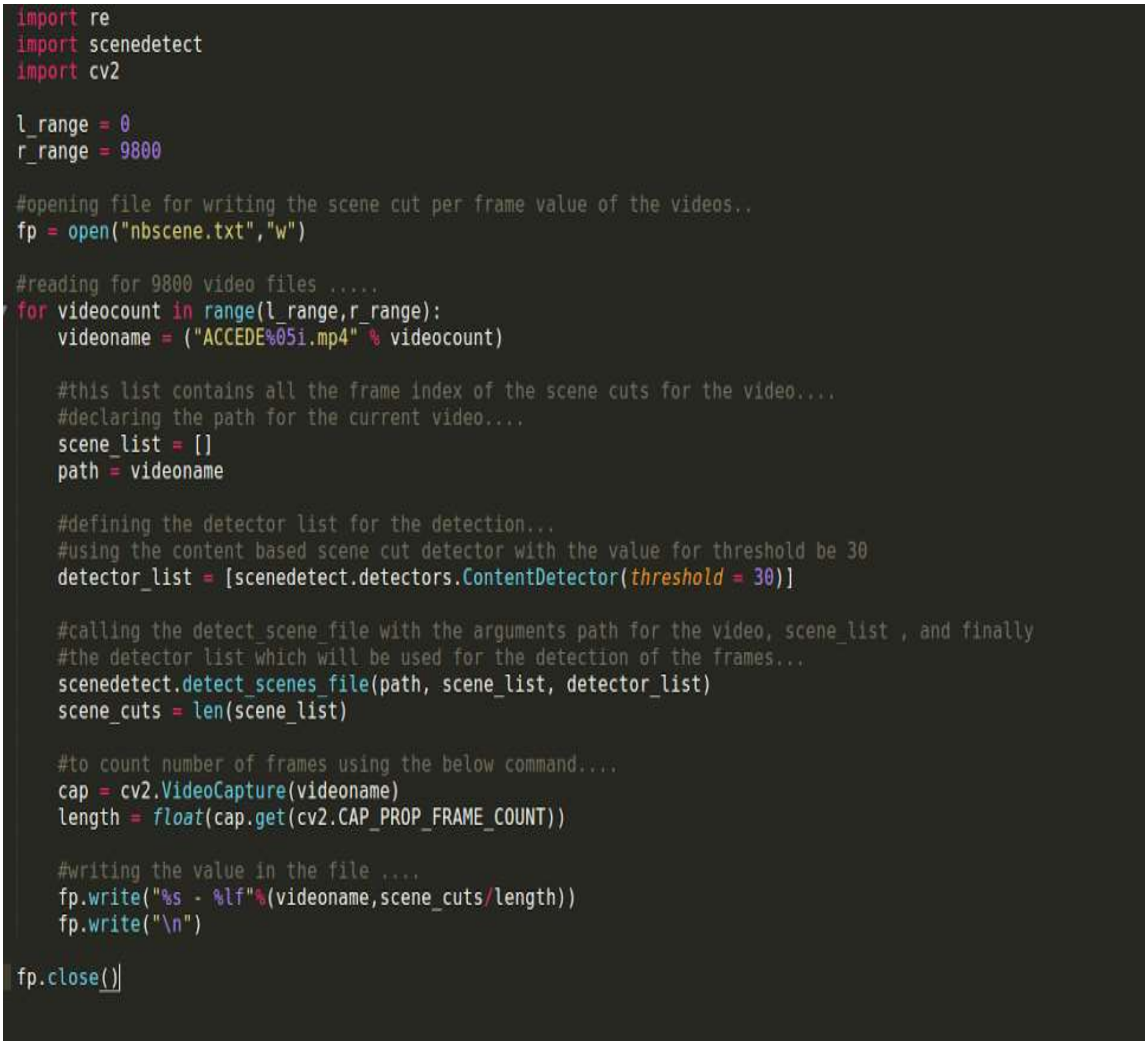
Whenever the director wants to get the viewer excited, he increases the scene cut rate. Similarly whenever the scene cut rate is low, the arousal induced is low.





| | Color | Motion<br>(Phase/intensity) | Shot cut rate |
|---|---|---|---|
| Fear | Dark and blue, sometimes dark and red<br>Low saturated | Zoom, tilt, dolly /NA | Fast |
| Sadness | Dark<br>Low Saturated | No camera motion / Small | Slow |
| Joy | Bright colors | NA<br>/ Large | NA |

## Algorithm Implemented -



```
import re
import scenedetect
import cv2

l_range = 0
r_range = 9800

#opening file for writing the scene cut per frame value of the videos..
fp = open("nbscene.txt","w")

#reading for 9800 video files .....
for videocount in range(l_range,r_range):
    videoname = ("ACCEDE%05i.mp4" % videocount)

    #this list contains all the frame index of the scene cuts for the video....
    #declaring the path for the current video....
    scene_list = []
    path = videoname

    #defining the detector list for the detection...
    #using the content based scene cut detector with the value for threshold be 30
    detector_list = [scenedetect.detectors.ContentDetector(threshold = 30)]

    #calling the detect_scene_file with the arguments path for the video, scene_list , and finally
    #the detector list which will be used for the detection of the frames...
    scenedetect.detect_scenes_file(path, scene_list, detector_list)
    scene_cuts = len(scene_list)

    #to count number of frames using the below command....
    cap = cv2.VideoCapture(videoname)
    length = float(cap.get(cv2.CAP_PROP_FRAME_COUNT))

    #writing the value in the file ....
    fp.write("%s - %lf"%(videoname,scene_cuts/length))
    fp.write("\n")

fp.close()
```

**Algorithm Explanation:**

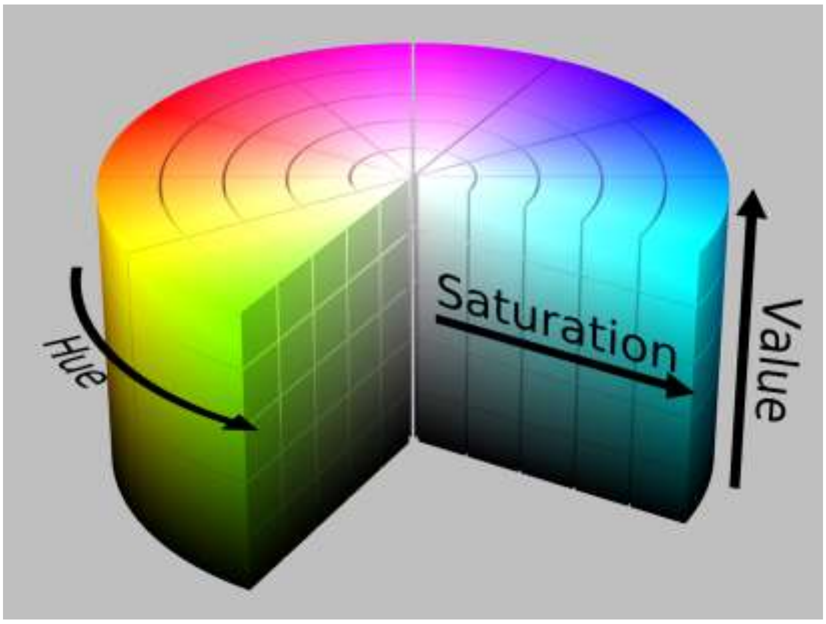
- ✓ To calculate the scene cuts per frame, we first need to calculate the number of scene cuts in the video.
- ✓ To calculate the scene cuts in the video, we use “PySceneDetect” library, which is a public library.
- ✓ To apply the method, we use content-Aware detector which uses the HSV color space with a threshold value of 30 and minimum scene length of 15 frames.
- ✓ To check if the two frames belong to same scene, it tries to find if the difference between two consecutive video frames is greater than threshold.
- ✓ If the difference is greater than the threshold then this frame is taken as the scene boundary.
- ✓ The length of the scene_list is taken as the number of scene cuts in the video.
- ✓ Along with the number of scene cuts, we also calculate the number of frames in the video.
- ✓ The scene cuts per frame value is proportional to number of scene cuts and inversely proportional to the number of frames.
- ✓ In the end (number of scene cuts/ number of frames) is taken as the number of scene cuts per frame.

## 5.4 Huecount

Hue count is a feature which counts the number of unique hues in the video. It gives a measure of colour vibrancy in the video. A higher number of hues generally implies that the video elicits a positive emotion.

**Algorithm:**

First we capture frames from the video. Next, these frames are converted from RGB(Red,Green,Blue) space to HSV(Hue,Saturation,Value). The HSV color space depicts colors based on a cylindrical model using cylindrical coordinates. A diagram of the representation is given below:



In the above cylinder,the angle along center axis gives hue, the distance from the axis gives value and the distance along the axis gives saturation. Thus we convert the images from RGB space to HSV space using the following formula.

**Algorithm for conversion from RGB to HSV:**

First the RGB values are divided by 255. Let the new values be R',G' and B'.

R' = R/255

G' = G/255

B' = B/255

Next we calculate the maximum and minimum of the R',G' and B' values as follows.

$C_{max}$ = max(R', G', B')

$C_{min}$ = min(R', G', B')

Δ = $C_{max}$ - $C_{min}$

Next we calculate the hue,saturation and value as follows:

**Hue:**

$$H = \begin{cases} 60^\circ \times \left(\frac{G'-B'}{\Delta} mod6\right) & ,Cmax = R' \\ 60^\circ \times \left(\frac{B'-R'}{\Delta} + 2\right) & ,Cmax = G' \\ 60^\circ \times \left(\frac{R'-G'}{\Delta} + 4\right) & ,Cmax = B' \end{cases}$$

**Saturation:**

$$S = \begin{cases} 0 & ,C_{max} = 0 \\ \frac{\Delta}{C_{max}} & ,C_{max} \neq 0 \end{cases}$$

**Value:**

V = $C_{max}$

Thus we have the HSV values.

**Algorithm for hue count:**

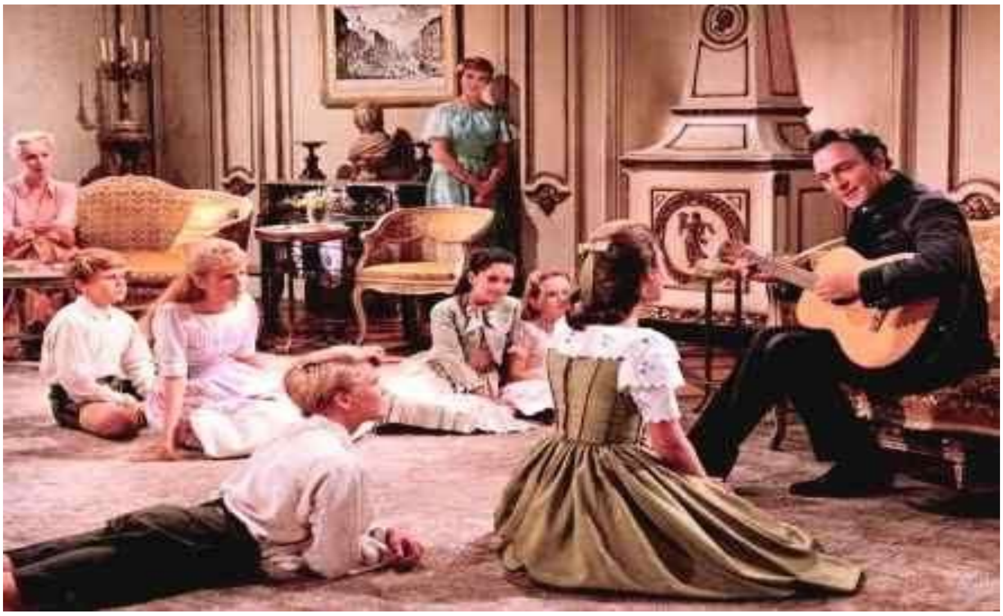
Using the HSV values derived above we calculate the number of unique hues for each frame. Next we take the average of the net number of unique hues in the video over the number of frames to get the average number of unique hues in each frame. This is the hue count of the video.

## 5.5 Lighting

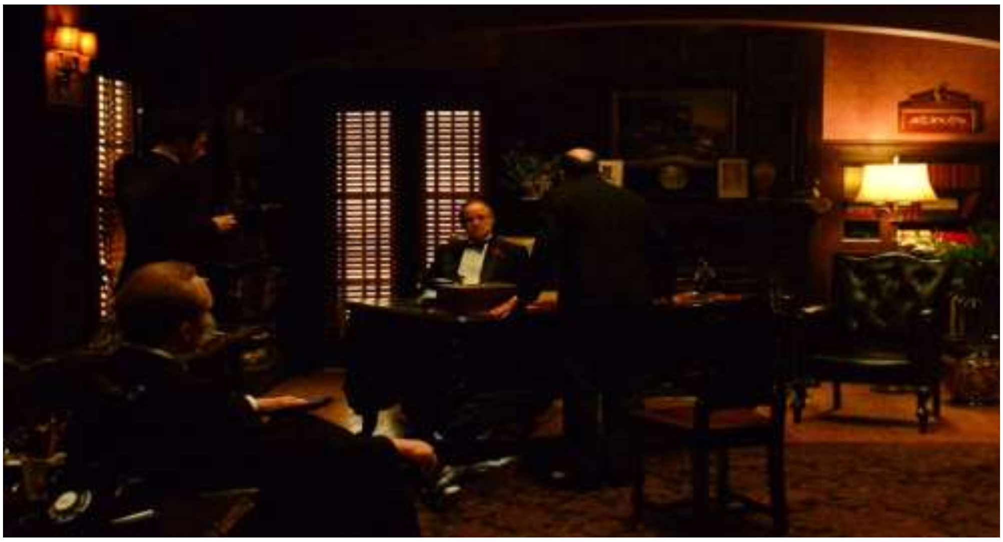
Lighting is a metric to measure the contrast in the luminescence of an image. It is a important feature to determine the affection of the scene. For an example, the scenes with positive emotions, that is involving pleasentness or joy are most often depicted with high luminescence, whereas those with negative feeling like sadness or horror contain low lighting value. It is a vital tool used by artists and film creators. Dramatic effects of scenes are enhanced by manipulating the intensity of light.

**Algorithm**:

For measuring the lightning metric, we followed the algorithm to measure the average intensity of the pixels in each frame of the video. In this algorithm, we first convert the BGR image to Grayscale image. Each pixel of grayscale image contains an intensity value of the respective pixel. We iterate though each pixel and find the sum of all the intensity values and then divide it by the total number of pixels in the image. This will give us the average intensity value of the image. Now we run this algorithm for every frame of the video to find the average luminescence of the video, with will give us the lightning metric.



A pleasing scene containing high luminescence.



Low luminescence value in a scene giving negative affection.

# 5.6 Depth of Field

Theoretically depth of field is a parameter which represents the distance between the farthest and nearest objects in a normal sharp image.
We classify depth of field into two categories: shallow depth of field and deep depth of field.
A scene is said to possess shallow depth of field if only a specific portion of that scene is highly focussed and rest of the area is blurred in the scene.
A scene is said to possess deep depth of field if for any two portion of image, difference in the focus magnitude is not much significant, i.e. the whole image appears equally focussed.

**Importance of Depth of field:**

DOF is an important part in cinematography and film field. It is a perfect tool to direct viewer's attention to any specific portion of the scene. In this way cinematographers assures that he doesn't let the viewer miss any minute detail such as some deep emotions of face,emphasizing or de-emphasizing on background,etc.

**Relation with emotion:**

Shallow Depth of field.

Deep Depth of field.

- Sometimes for simply expressing the emotional state of a character,the effect of change in DOF can be handy to show emotional state of subject in scene.
- In terms of emotion,depth of field is a parameter of valence.
- A shallow depth of field signifies negative valence like (tension,sad,stressed), whereas a deep depth of field signifies positive valence like (calm,relaxed).
- Shallow depth of field signifies psychological introspection.
- Depth of field express subject’s struggle to express an enhanced emotional state of mind.

For example: if the scene contains a man as a main subject and if he is highly focussed then the viewer can get a tensed affection,etc.

**Algorithm for DOF calculation :**

To determine the DOF, the algorithm which we used has the following argument as basis - “there is a change in the vertical and horizontal derivative histogram after we have performed the blurring using a fixed size kernel”.

1. To blur image firstly we choose a blurring kernel window [k x k] with blurring coefficients equal to 1/(K*K).

2. Here fk denotes the blurring kernel window [k x k]. We convolve $f_k$ with I (image) and to compute the vertical and horizontal derivatives from I * $f_k$, we further convolve it with dx ([1,-1]) and

dy([{1},{-1}]) respectively to get the vertical and horizontal derivatives :

$$p_{xk} \quad \alpha \quad hist(I * f_k * d_x)$$

$$p_{yk} \quad \alpha \quad hist(I * f_k * d_y)$$

3. Now choose pixel(i,j) and a blurring kernel K and then calculate the K-L divergence between the distribution ($px_k$,$py_k$) and ($px_1$ and $py_1$) as follows :

$$KL(p|q)(i,j) = p_{ij} log(\frac{p_{ij}}{q_{ij}})$$

4. Then calculate Dk as follows :

$$D_k(i,j) = \sum_{(n,m)\in W_{ij}} KL(p_{xk}|p_{x1})(n,m) + KL(p_{yk}|p_{y1})(n,m)$$

Dk is close to zero when pxk and pyk are close in magnitude to $px_1$ and $py_1$.

Which further means the image under test is insensitive to blur , by which we

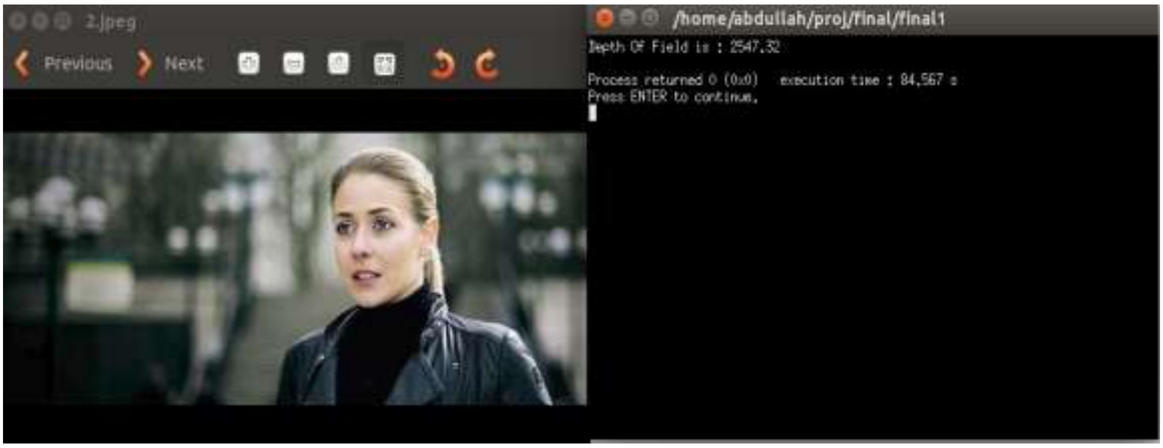
Conclude that the image is already blurred.

5. The value of DOF is calculated using the following formula :

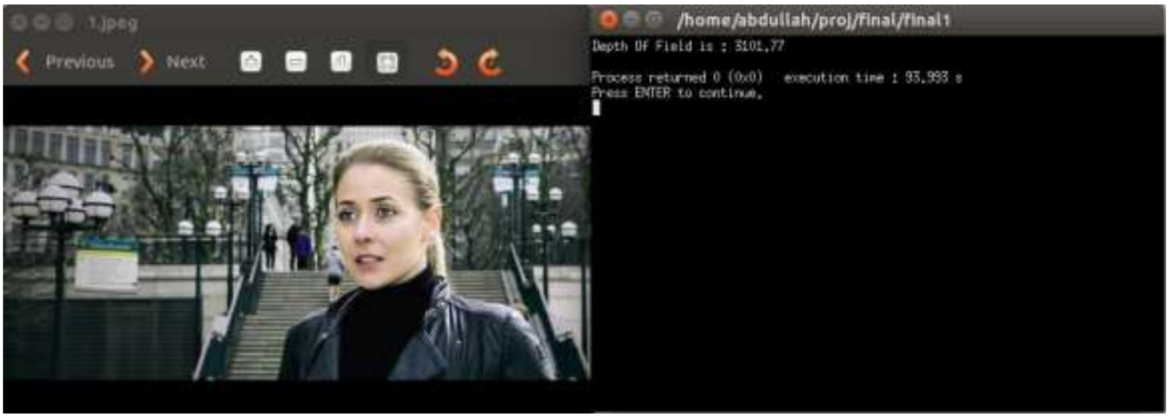
$$DoF = \sum_{(i,j)\in I} \sum_{k} D_k(i,j)$$

Lower DOF signifies that image is highly blurred and higher value shows that image posses deep DOF.

## Result details/snippets :



```
Depth Of Field is : 2547.32

Process returned 0 (0x0)   execution time : 84.567 s
Press ENTER to continue.
```



```
Depth Of Field is : 3101.77

Process returned 0 (0x0)   execution time : 93.993 s
Press ENTER to continue.
```

## 5.7 Global Activity

Global Activity is a feature which captures the average amount of motion in a video. The apparent motion in a video can be captured using motion vectors. These motion vectors can be approximated by calculating the optical flow in a video.

Optical flow or optic flow represents the apparent motion of objects in a visual scene which occurs due to the relative motion between an observer (the camera) and the scene.

We have used the Gunnar Farneback's Algorithm to calculate the dense optical flow in the video. Dense optical flow calculates the motion in every section of the frame.

**Gunnar Farneback's Algorithm:**

This algorithm gives a displacement field for two successive frames. In this algorithm we do polynomial expansion of the image. Through polynomial expansion we aim to approximate a neighborhood of a point using a 2D function with a polynomial. Considering quadratic polynomial basis as 1, x2, y2, x, y, xy, values of pixel in a neighborhood of image is represented by:

$$f(x)\; x^T Ax + b^T x + c$$

Where A is a symmetric matrix, b is a vector and c is a scalar.
Now if we consider a displacement of d at point x, the new polynomial is:

$$\begin{aligned}
f_1(x) &= x^T A_1 x + b_1^T x + c_1 \\
f_2(x) = f_1(x-d) &= (x-d)^T A_1 (x-d) + b_1^T (x-d) + c_1 \\
f_2(x) = f_1(x-d) &= (x)^T A_1 (x) + (b_1 - 2A_1 d)^T (x) \\
&\quad + d^T A d - b_1^T d + c_1
\end{aligned}$$

Equating Coefficients we get:

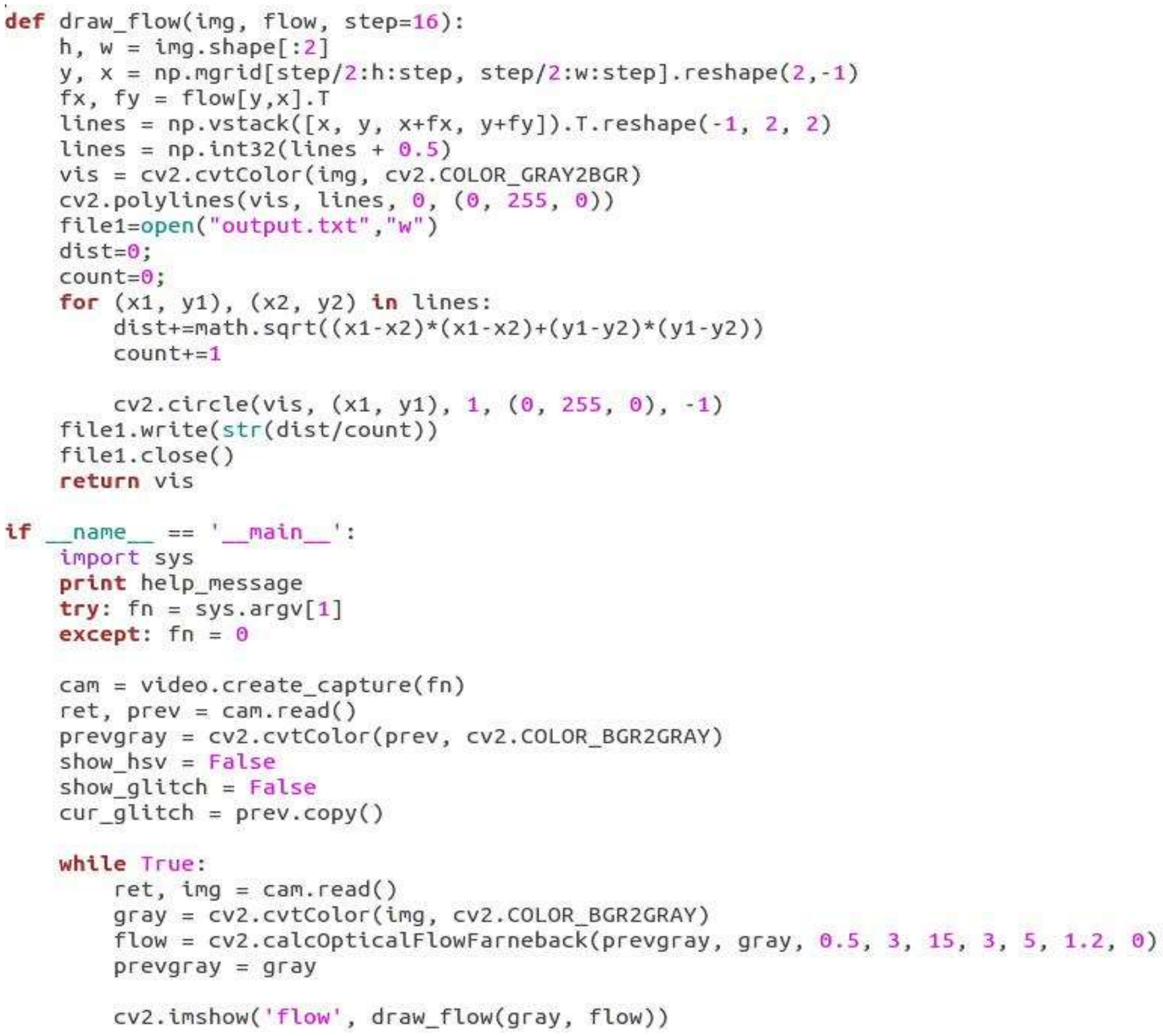
$$\begin{aligned} A_2 &= A_1 \\ b_2 &= (b_1 - 2A_1 d) \\ c_2 &= d^T A d - b_1^T d + c_1 \end{aligned}$$

Assuming A is non-singular:

$$d = -\frac{1}{2} A^{-} 1(b_2 - b_1)$$

Thus by equating the coefficients of the polynomial thedisplacement vector is obtained at sections in the image assuming there is overlap between the region of interest i.e. image neighborhood in adjacent frames.

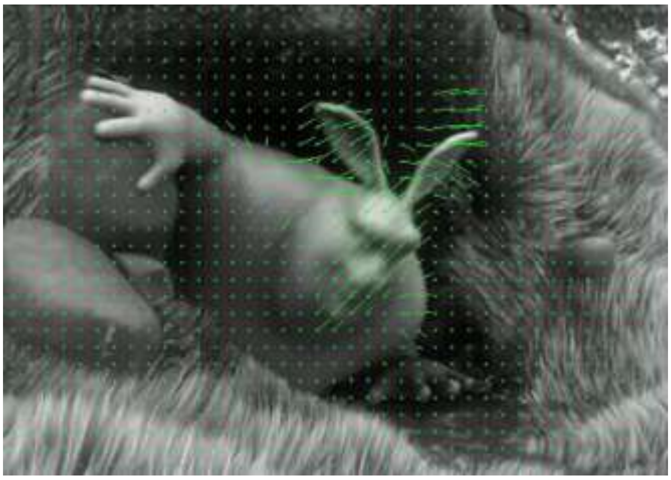
**Code Snippet:**



```
def draw_flow(img, flow, step=16):
    h, w = img.shape[:2]
    y, x = np.mgrid[step/2:h:step, step/2:w:step].reshape(2,-1)
    fx, fy = flow[y,x].T
    lines = np.vstack([x, y, x+fx, y+fy]).T.reshape(-1, 2, 2)
    lines = np.int32(lines + 0.5)
    vis = cv2.cvtColor(img, cv2.COLOR_GRAY2BGR)
    cv2.polylines(vis, lines, 0, (0, 255, 0))
    file1=open("output.txt","w")
    dist=0;
    count=0;
    for (x1, y1), (x2, y2) in lines:
        dist+=math.sqrt((x1-x2)*(x1-x2)+(y1-y2)*(y1-y2))
        count+=1

        cv2.circle(vis, (x1, y1), 1, (0, 255, 0), -1)
    file1.write(str(dist/count))
    file1.close()
    return vis

if __name__ == '__main__':
    import sys
    print help_message
    try: fn = sys.argv[1]
    except: fn = 0

    cam = video.create_capture(fn)
    ret, prev = cam.read()
    prevgray = cv2.cvtColor(prev, cv2.COLOR_BGR2GRAY)
    show_hsv = False
    show_glitch = False
    cur_glitch = prev.copy()

    while True:
        ret, img = cam.read()
        gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
        flow = cv2.calcOpticalFlowFarneback(prevgray, gray, 0.5, 3, 15, 3, 5, 1.2, 0)
        prevgray = gray

        cv2.imshow('flow', draw_flow(gray, flow))
```

**Explanation of Above Code:**

In the above code, we are first capturing frames from the video. Next we are convert the frame from RGB to GRAYSCALE. After that, we are using the Farneback optical flow function to calculate displacement in each successive frames. We take the average of net displacement for the whole video over number of frames to get Global Activity.



Calculating Dense Optical Flow

## 5.8 Length of Scene Cuts

Whetherthe scenestays on for long or ends Shortly, length of a scene cutconveys something to the viewer.

**Holding Long**

Lingering scene depicts an emotional as well as spatial meaning to the scene as well.

Emotionally, this kind of lingering shot is interesting because:

1) Slow pacing depicts the despondency who has arrived to a decision

2) Slow cutting shows that slow flickering of that despondency

Longer scene builds tension, suspence and anticipation.

**Cutting Short**

Length is propotional to pacing of the scene. Shorter the scene, faster the pacing. And faster the pacing, more energy is into that scene. So anger, excitement, happy can be judged on the basis of the length.

Shorter scenes break the rythym and devoids the scene with meaning and instead give a feeling of anxiety and restlessness. Or it can depict a lot of motion in that scene, like 2 people arguing.

**Algorithm Explanation:**

- ✓ To calculate the length of scene cuts, we first need to calculate the number of scene cuts in the video.
- ✓ To calculate the length of scene in the video, we use “PySceneDetect” library, which is a public library.
- ✓ To apply the method, we use content-Aware detector which uses the HSV color space with a threshold value of 30 and minimum scene length of 15 frames.
- ✓ To check if the two frames belong to same scene, it tries to find if the difference between two consecutive video frames is greater than threshold.
- ✓ If the difference is greater than the threshold then this frame is taken as the scene boundary.
- ✓ The length of the scene_list is taken as the number of scene cuts in the video.
- ✓ Along with the number of scene cuts, we also calculate the number of frames in the video.
- ✓ Each scene is composed of a starting fram and an ending frame.
- ✓ In the end Length between each first frame is taken as the Length of scene.

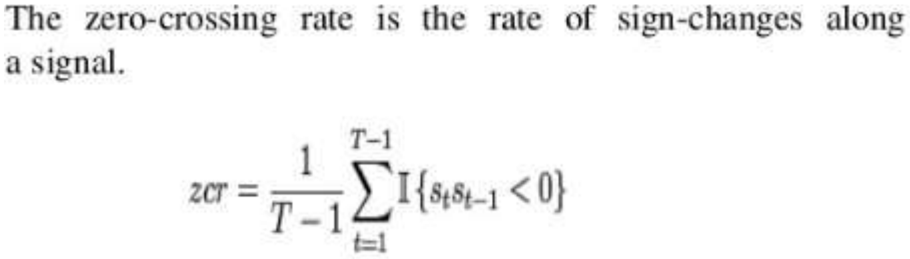
## 5.9 Zero Crossing Rate

The **zero-crossing rate** is the **measure** of changes in signof signal. The frequency or how fast the signal changes to and fro from positive to negetive and back.

In Music info extraction and Speech recognition, ZCR has been used consistently. It can also be used to classify sounds.

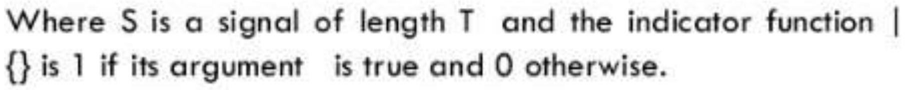
It also used to detect weather a human is present in the scene or not.

A reasonable generalization is that if the ZCR value is high the speech signal is unvoiced. While if the ZCR is low the speech signal is voiced. This is mainly because high frequencies imply high ZCR and there is a strong correlation between ZCR and energy distribution of the signal.



The zero-crossing rate is the rate of sign-changes along a signal.

$$zcr = \frac{1}{T-1}\sum_{t=1}^{T-1} \mathrm{I}\{s_t s_{t-1} < 0\}$$



Where S is a signal of length T and the indicator function | {} is 1 if its argument is true and 0 otherwise.

**Algorithm :**

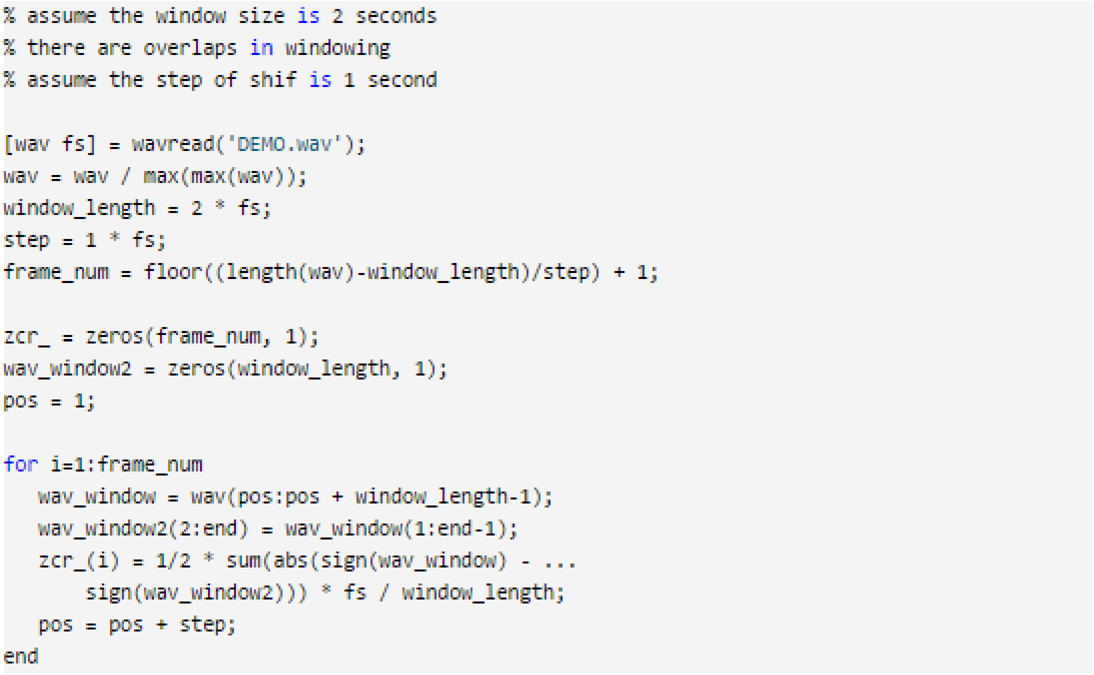
1. First the mp4 video file has to be converted into a wav file. i.e. audio has to be extracted from it.

2. Then the audio signal is sampled with a window of 2 seconds and shifted by 1 sec each time.

3. Then over the frames, the formulae is applied and the final zcr is derived.

4. For calculating zero crossing rate, I used a open sourced public library called Yaafe library.

## Code Snippet



```
% assume the window size is 2 seconds
% there are overlaps in windowing
% assume the step of shif is 1 second

[wav fs] = wavread('DEMO.wav');
wav = wav / max(max(wav));
window_length = 2 * fs;
step = 1 * fs;
frame_num = floor((length(wav)-window_length)/step) + 1;

zcr_ = zeros(frame_num, 1);
wav_window2 = zeros(window_length, 1);
pos = 1;

for i=1:frame_num
   wav_window = wav(pos:pos + window_length-1);
   wav_window2(2:end) = wav_window(1:end-1);
   zcr_(i) = 1/2 * sum(abs(sign(wav_window) - ...
       sign(wav_window2))) * fs / window_length;
   pos = pos + step;
end
```

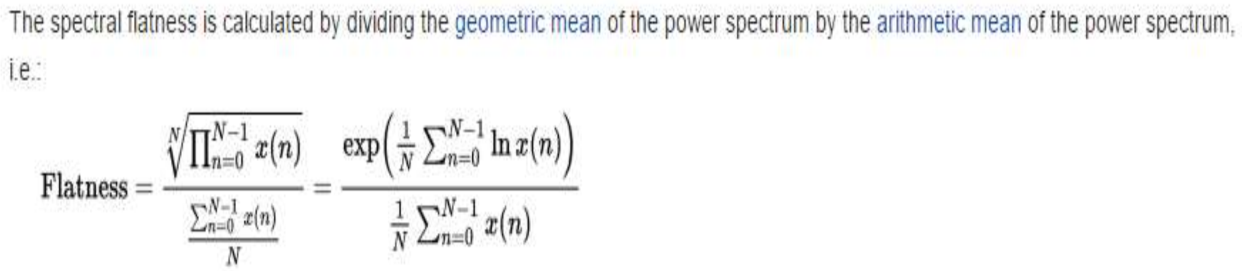
# 5.10 Spectral Flatness

Used in digital signal processing, it is used to measure the audio or signal spectrum.

The biggest feature why we use this to quantify emotion is that it helps us in determing how tonal or noisy the sound is.

Each signal has a power spectrum. And from the shape of the spectrum, we can see the peaks and crests. Or in other words, we can judge the flatness of the signal.

Now, more flat the signal, more tonal the sound is and more calm and sad it will be percieved to be.
More peaks and crests mean more noice, or excitement or anger.



The spectral flatness is calculated by dividing the geometric mean of the power spectrum by the arithmetic mean of the power spectrum, i.e.:

$$\text{Flatness} = \frac{\sqrt[N]{\prod_{n=0}^{N-1} x(n)}}{\frac{\sum_{n=0}^{N-1} x(n)}{N}} = \frac{\exp\left(\frac{1}{N}\sum_{n=0}^{N-1} \ln x(n)\right)}{\frac{1}{N}\sum_{n=0}^{N-1} x(n)}$$

This is then converted to a Db or decibal range.

**Algorithm**:

1. We just need to measure the above stated formulae over the frames or windows of the digital signal.

2. First we convert the mp4 video file to a wav signal.

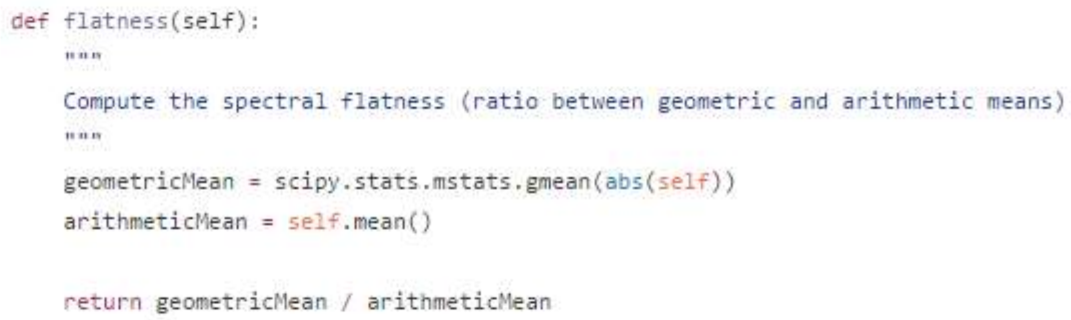
3. Frame’s length on which perform FFT. Original frame is padded with zeros or truncated to reach this size. If 0 then use original frame length.

4. The frame size is kept 2 sec and the skip is of 1 sec. Weighting window to apply before fft. Hanning or Hamming.

5. Then in the frames, I use the formulae to get the geometric and arithmetic mean and take the ratio.

6. The ratio is the spectral flatness measure.

**Code Snipett of the function:**



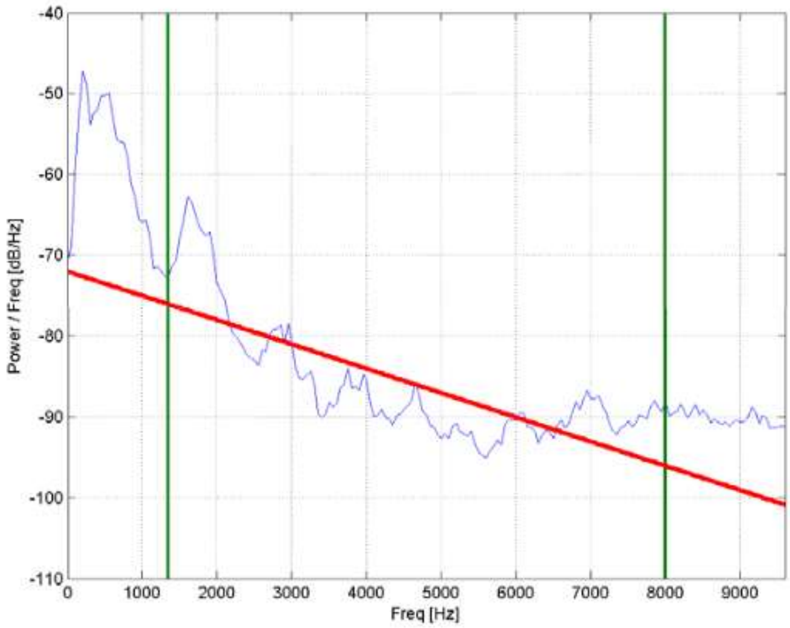
```
def flatness(self):
    """
    Compute the spectral flatness (ratio between geometric and arithmetic means)
    """
    geometricMean = scipy.stats.mstats.gmean(abs(self))
    arithmeticMean = self.mean()

    return geometricMean / arithmeticMean
```

# 5.11 Spectral Slope

Spectral tilt, or spectral slope, is an important parameter in voice synthesis and voice perception. It is a measure of voice quality.Those voice quality include harsh, tense, breathy etc.

The value of the spectral slope represents the amount of decrease of the spectralamplitude. It shows how rapidly the amplitudes of successive partials (component frequencies) decrease as the value of the frequency increases.

A less steep spectral slope represents to a louder voice and when the loudness of the audio decreases, the spectral slope increases.



**Algorithm:**

1. First we convert the mp4 video file to a wav signal.

2. Frame's length on which perform FFT. Original frame is padded with zeros or truncated to reach this size. If 0 then use original frame length.

3. The frame size is kept 2 sec and the skip is of 1 sec. Weighting window to apply before FFT Hanning or Hamming.

To quantify this is by applying linear regression to the spectrum of the signal, which produces a single number indicating the slope of the line of best fit through the spectral data.
To calculate the video we use the Yaafe Library, which is a public library.

# 6. Classification and Clustering Algorithm Used

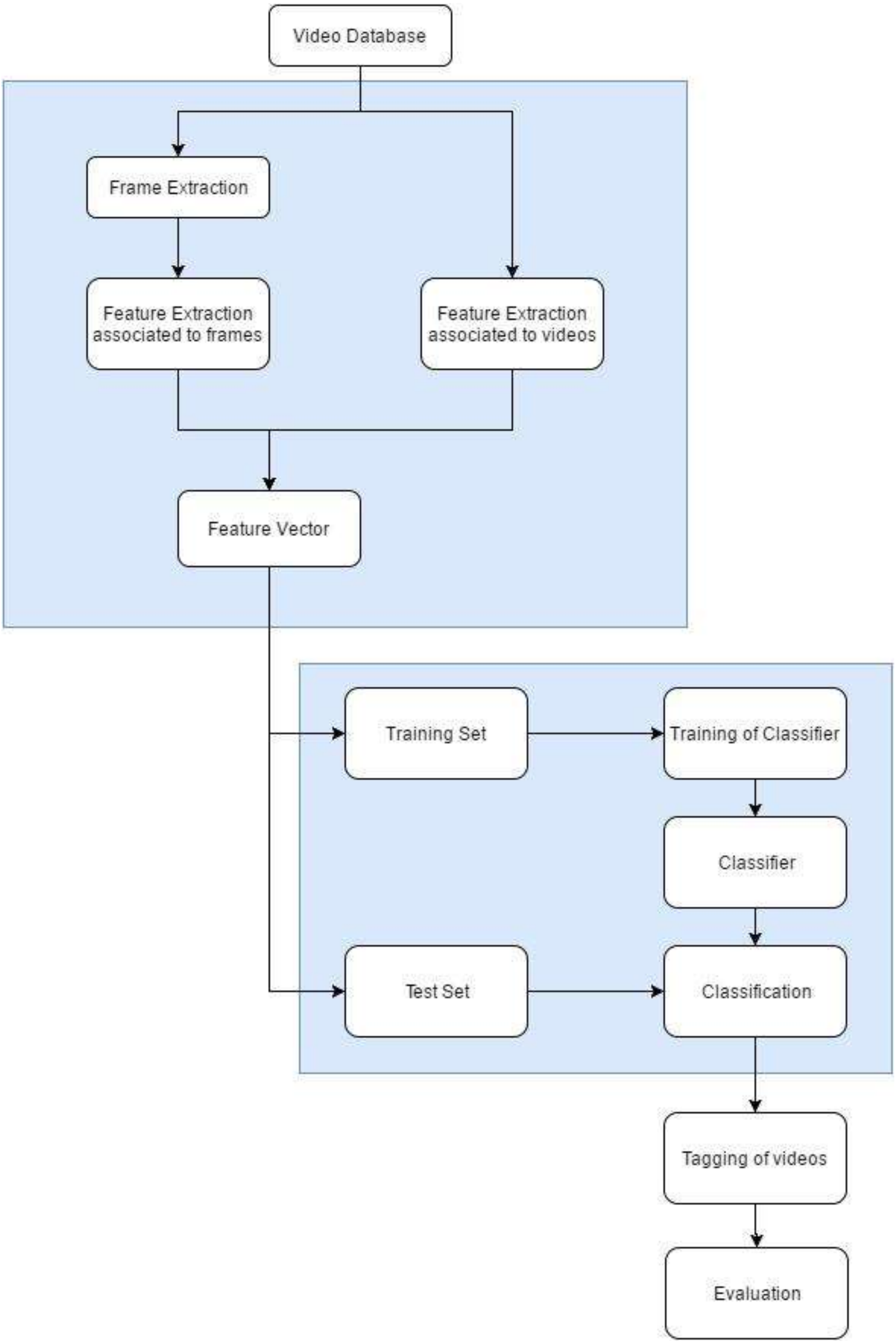
## Clustering:

We have used Kmeans algorithm for clustering data based on valence arousal values. K-means aims to partition n observations into k clusters in which each observation belongs to the cluster with the nearest mean serving as a prototype of the cluster. This results in partitioning of data space into Voronoi cells.

## Classification:

We have used KNN(K Nearest Neighbours) algorithm for classification. KNN is a non-parametric method used for classification and regression in which input consists of k-closest training examples in a space and output is a class membership. An object is classified by a majority vote of its neighbours, with the object being assigned to the class most common among its K-nearest neighbours.

# 7.Methodology



## Steps:

- Clustering has been done on the valence-arousal values using K-means algorithm.

- Then we partition the data in a 70-30 ratio for training and classification.

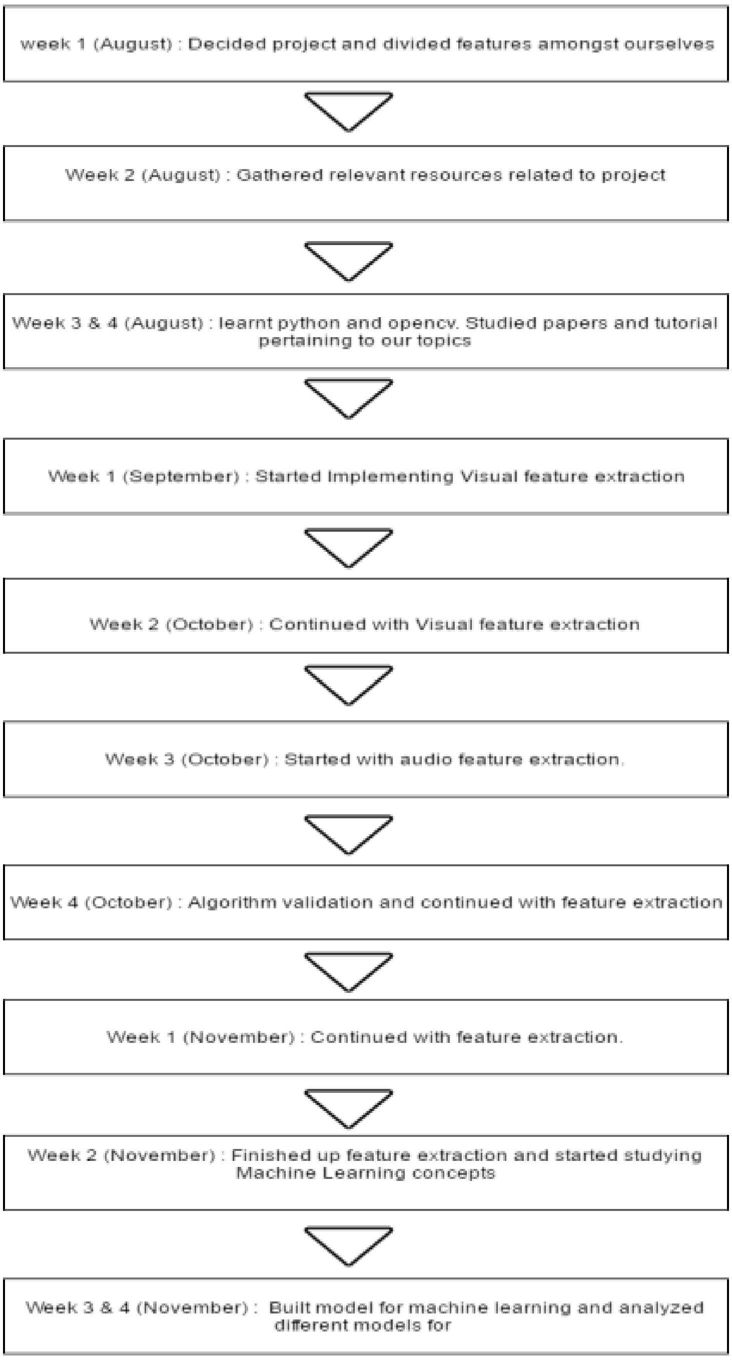
- We use KNN algorithm for classification.

- Then we calculated confusion tables and accuracies.

# 8. Tools Required

- Python 2.7
- OpenCV Library
- Any C++ code editor and C++ compiler

# 9.Work Flow



week 1 (August) : Decided project and divided features amongst ourselves

Week 2 (August) : Gathered relevant resources related to project

Week 3 & 4 (August) : learnt python and opencv. Studied papers and tutorial pertaining to our topics

Week 1 (September) : Started Implementing Visual feature extraction

Week 2 (October) : Continued with Visual feature extraction

Week 3 (October) : Started with audio feature extraction.

Week 4 (October) : Algorithm validation and continued with feature extraction

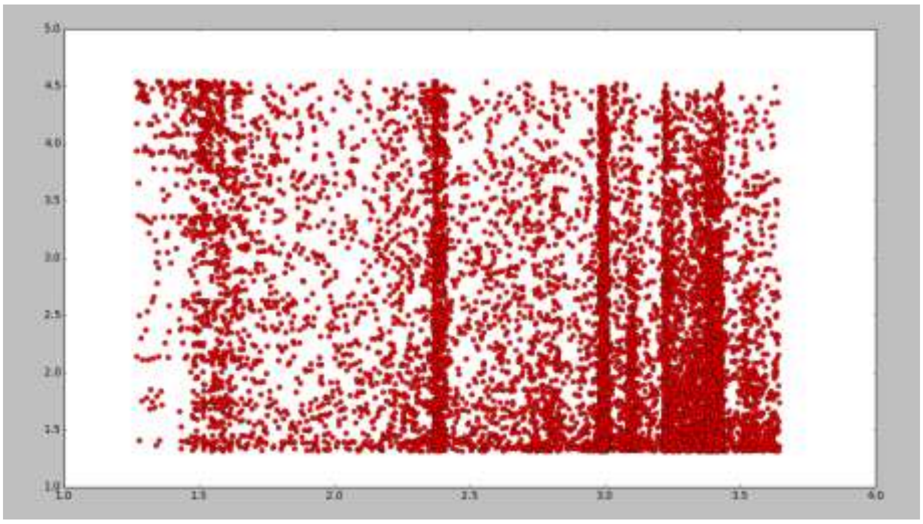
Week 1 (November) : Continued with feature extraction.

Week 2 (November) : Finished up feature extraction and started studying Machine Learning concepts

Week 3 & 4 (November) : Built model for machine learning and analyzed different models for

# 10. Results

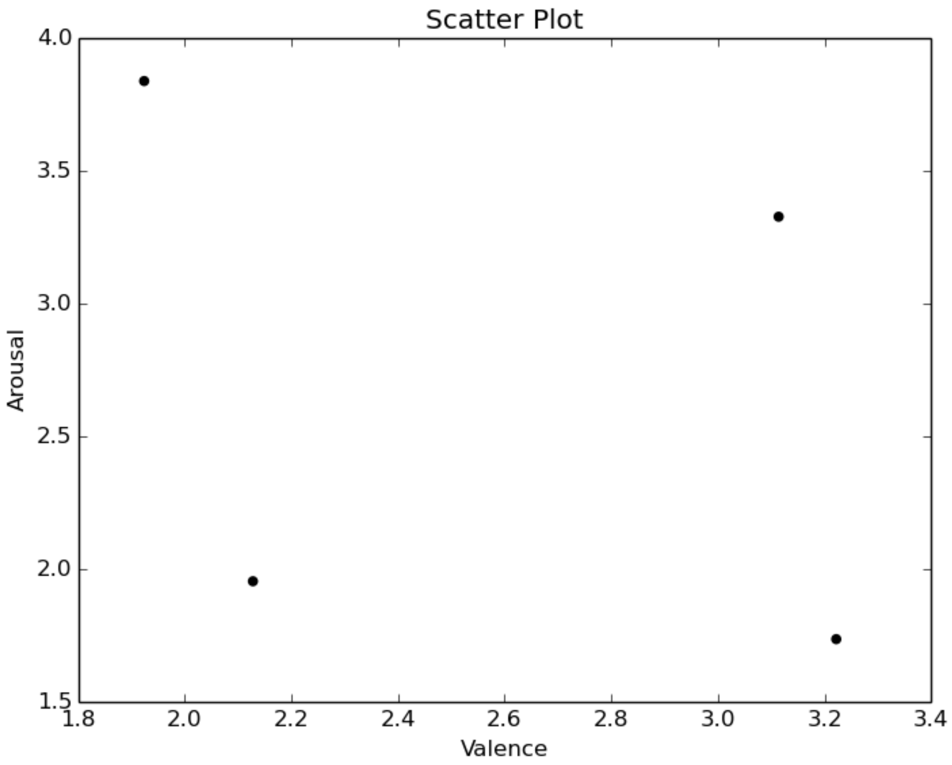
- **Based on the arousal valence values we get the following plot.**

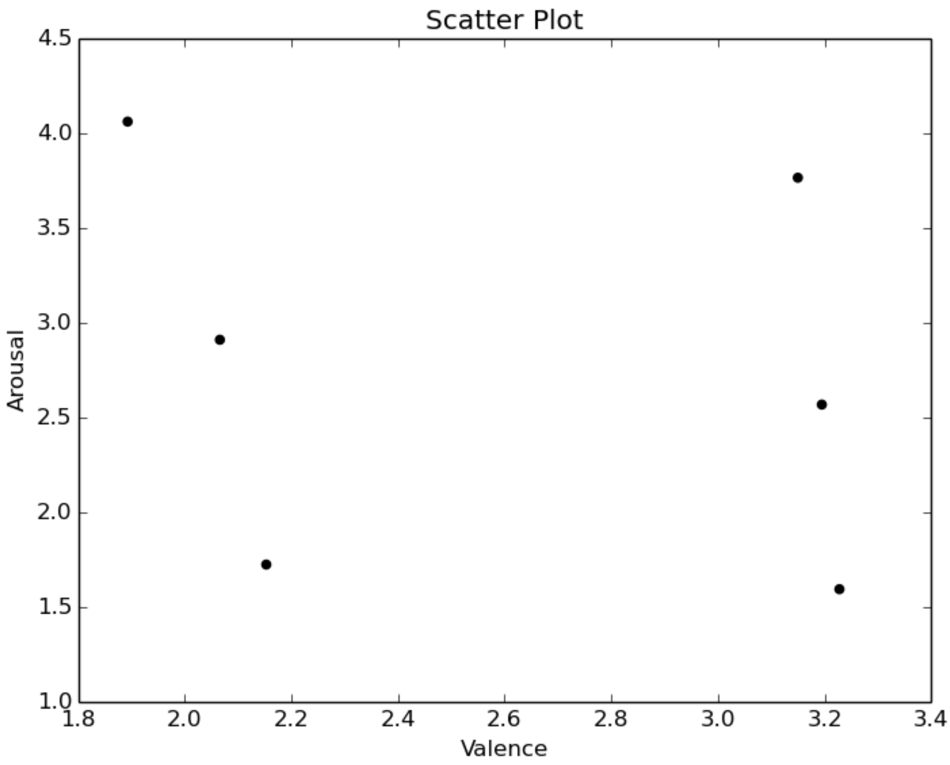


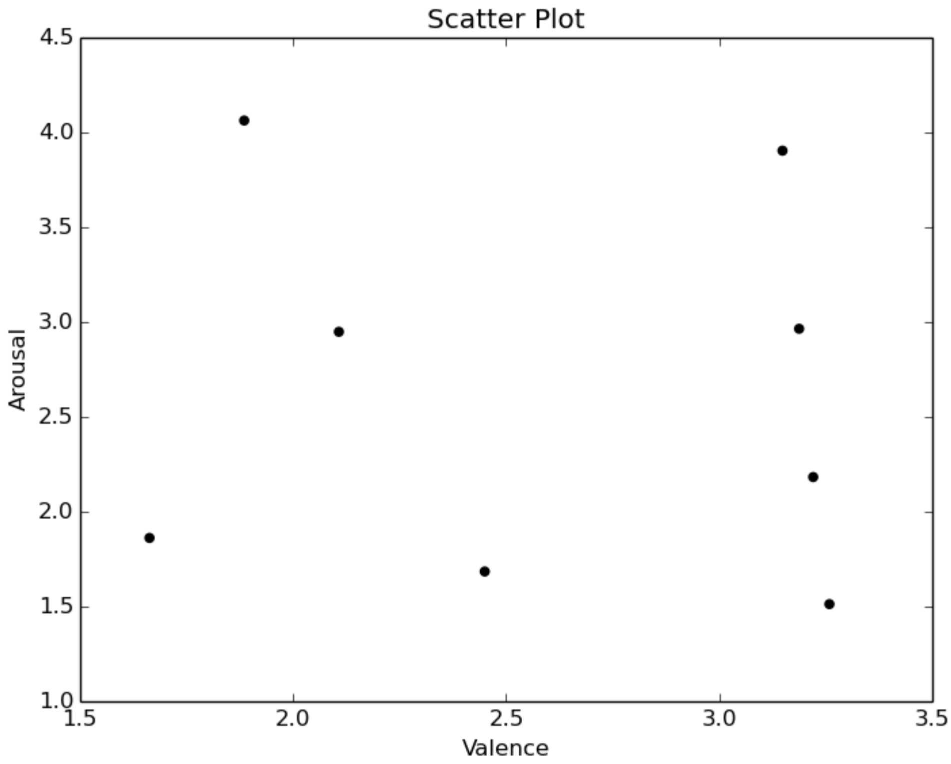
- **Classifying based on the tagged data we get the following confusion table. The total accuracy for this is 54.71%.**

| | Happy / Excited | Sad /Tense | Upset/Distressed | Calm / Relaxed | Accuracy (%) |
|---|---|---|---|---|---|
| Happy / Excited | 243 | 31 | 49 | 273 | 40.77 |
| Sad /Tense | 34 | 214 | 16 | 130 | 54.04 |
| Upset/Distressed | 36 | 9 | 229 | 148 | 54.26 |
| Calm / Relaxed | 271 | 91 | 180 | 846 | 60.95 |

- Next to identify videos based on valence and arousal values, we have clustered them based on arousal-valence using Kmeans algorithm for k=4,6 and 8 clusters. The cluster centers for each clustering are depicted in the following pictures.




Scatter Plot
Arousal
Valence
4.5
4.0
3.5
3.0
2.5
2.0
1.5
1.0
1.8
2.0
2.2
2.4
2.6
2.8
3.0
3.2
3.4


Scatter Plot
Arousal
Valence
4.5
4.0
3.5
3.0
2.5
2.0
1.5
1.0
1.5
2.0
2.5
3.0
3.5

- After that we have classified the test videos using KNN algorithm. It generates the following three confusion tables. The accuracies for each of these clusterings are **43.14%, 33.46%** and **24.64%**.

| | 1 | 2 | 3 | 4 | Accuracy(%) |
|---|---|---|---|---|---|
| 1 | 452 | 92 | 103 | 67 | 63.4 |
| 2 | 96 | 178 | 82 | 48 | 44.05 |
| 3 | 278 | 309 | 384 | 224 | 32.13 |
| 4 | 104 | 54 | 116 | 194 | 38.95 |

| | 1 | 2 | 3 | 4 | 5 | 6 | Accuracy (%) |
|---|---|---|---|---|---|---|---|
| 1 | 166 | 61 | 34 | 124 | 96 | 77 | 29.74 |
| 2 | 86 | 394 | 113 | 98 | 147 | 93 | 42.31 |
| 3 | 113 | 81 | 106 | 134 | 93 | 55 | 18.21 |
| 4 | 39 | 60 | 70 | 42 | 16 | 43 | 15.55 |
| 5 | 27 | 41 | 49 | 76 | 37 | 49 | 13.26 |
| 6 | 39 | 61 | 43 | 27 | 17 | 192 | 51.23 |

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | Accuracy (%) |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 18 | 27 | 13 | 26 | 33 | 9 | 14 | 2 | 14.75 |
| 2 | 71 | 93 | 63 | 31 | 42 | 22 | 42 | 93 | 20.35 |
| 3 | 53 | 41 | 23 | 88 | 81 | 45 | 91 | 131 | 32.03 |
| 4 | 30 | 22 | 34 | 73 | 18 | 29 | 21 | 30 | 28.40 |
| 5 | 41 | 35 | 31 | 29 | 73 | 39 | 52 | 21 | 22.74 |
| 6 | 34 | 41 | 27 | 37 | 29 | 120 | 31 | 39 | 33.42 |
| 7 | 42 | 28 | 35 | 22 | 44 | 28 | 26 | 53 | 9.35 |
| 8 | 41 | 25 | 45 | 33 | 35 | 19 | 33 | 54 | 18.94 |

- After that we have classified the test videos using KNN algorithm. It generates the following three confusion tables. The accuracies for each of these clusterings are **42.96% ,** **33.35%** and **24.64%.**

| | 1 | 2 | 3 | 4 | Accuracy(%) |
|---|---|---|---|---|---|
| 1 | 31 | 2 | 675 | 6 | 4.34 |
| 2 | 17 | 0 | 377 | 3 | 0 |
| 3 | 22 | 1 | 1171 | 3 | 97.8 |
| 4 | 17 | 1 | 473 | 1 | 0.20 |

| | 1 | 2 | 3 | 4 | 5 | 6 | Accuracy (%) |
|---|---|---|---|---|---|---|---|
| 1 | 22 | 518 | 7 | 2 | 0 | 9 | 3.94 |
| 2 | 23 | 902 | 0 | 3 | 0 | 3 | 96.88 |
| 3 | 8 | 363 | 7 | 2 | 0 | 2 | 1.8 |
| 4 | 6 | 258 | 3 | 1 | 0 | 2 | 0.37 |
| 5 | 11 | 261 | 2 | 2 | 0 | 3 | 0 |
| 6 | 12 | 354 | 3 | 3 | 0 | 2 | 0.53 |

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | Accuracy (%) |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 0 | 7 | 106 | 2 | 0 | 1 | 3 | 3 | 0 |
| 2 | 0 | 39 | 393 | 5 | 6 | 10 | 1 | 3 | 8.5 |
| 3 | 0 | 52 | 633 | 11 | 7 | 4 | 2 | 12 | 87.79 |
| 4 | 0 | 17 | 226 | 4 | 2 | 2 | 2 | 4 | 1.55 |
| 5 | 0 | 24 | 270 | 7 | 3 | 5 | 2 | 10 | 1.02 |
| 6 | 1 | 25 | 312 | 4 | 5 | 2 | 5 | 5 | 0.5 |
| 7 | 0 | 15 | 250 | 5 | 2 | 2 | 0 | 4 | 0 |
| 8 | 0 | 19 | 243 | 5 | 3 | 4 | 2 | 9 | 3.15 |

# 8. Conclusion

- We achieved an accuracy of 54.71% on supervised learning on the tagged data.
- We clustered the data on the basis of Valence and Arousal values of the crowd sourced data, for cluster size of 4, 6 and 8. On applying classification on these models, we achieved lesser accuracy of 43.14%, 33.46% and 24.64% respectively.

# 9. FURTHER WORKS

- Find the goodness of the features.
- Optimise the features extracted and incorporate more features helping us improve the efficiency and accuracy.
- Apply better classification algorithms for the affectie analysis of the data.
- Develop an application based on our project for day to day use and develop systems to use in online video websites and search and retrieval systems.

# 12. References

[1] O. Le Meur, T. Baccino, and A. Roumy, “Prediction of the inter-observer visual congruency (IOVC) and application to image ranking,” in Proceedings of the 19th ACM International Conference on Multimedia, 2011, pp. 373–382.

[2] Y. Luo and X. Tang, “Photo and video quality evaluation: Focusing on the subject,” in Proceedings of the 10th International Conference on Computer Vision, 2008, vol. 5304, pp. 386–399

[3] Yoann Baveye, Emmanuel Dellandrea, Christel Chamaret and Liming Chen, “LIRIS-ACCEDE: A Video Database for Affective Content Analysis”

[4] D. Hasler and S. Suesstrunk, “Measuring colourfulness in natural images,” in Proc. SPIE Electronic Imaging 2003: Human Vision and Electronic Imaging VIII, vol. 5007, 2003, pp. 87–95.

[5] Naz Kaya, "Relationship between color and emotion: a study of college students"

[6] Hang-Bong Kang, Dept. of Computer Engineering, The Catholic University of Korea, "Affective Content Detection using HMMs"

[7] Virginia Fern´andez Arguedas, "Affective Video Content Representation"

[8] Gunnar Farneback, “Two-Frame Motion Estimation Based on Polynomial Expansion”

[9] Gunnar Farneback, “Polynomial Expansion for Orientation and Motion Estimation”

[10]Hasler ,David, Sabine E. Suesstrunk and Thrasyvoulos N. Pappas. “Human Vision And Electronic Imaging VIII”

[11]Nikhil Sawant ,“ Color Harmonization for videos”

[12]Jones, Gareth J.F., and Ching Hua Chan, “Affect based indexing for multimedia data”.