Facial Expression Recognition using Saliency maps and information computation

---

Abhinav Agarwal 2013A7PS124P

on

10 May 2016

BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE, PILANI (Rajasthan)

ACKNOWLEDGEMENTS

Every project big or small is successful largely due to the effort of a number of wonderful people who have always given their valuable advice or lent a helping hand. I sincerely appreciate the inspiration; support and guidance of all those people who have been instrumental in making this project a success.

I, a student of **Birla Institute of Technology And Science, Pilani**, am extremely grateful to "**Central Electronics Engineering Research Institute, Pilani, Rajasthan**" for the confidence bestowed in me and entrusting me with project entitled "**Facial Expression Recognition using saliency maps and information computation**".

At this juncture I feel deeply honoured in expressing our sincere thanks to

Dr. Atanendu Sekhar Mandal
Chief Scientist, Perception Engineering Lab
CEERI Pilani

for making the resources available at right time and providing valuable insights leading to the successful completion of my project.

Abhinav Agarwal
2013A7PS124P
10 May 2016

CONTENTS

Introduction to expression recognition

Humans express emotions in daily interaction. They are channelled through our voice to express feelings or likings and reflected in our face, hands and other body gestures. Research has shown that the most expressive way humans display emotions is through facial expression. It has also been theorized through research that the verbal part of a message contributes only for 7%, the vocal part for 38%, while facial expressions for 55% for the whole of the effect of speaker's message. Emotions are an integral part of our existence, as we smile to show greeting, frown in confusion, shout, widen our eyes or chatter our teeth in rage. They are our feeling or response to a particular situation or environment. Our communication and reactions are enriched by our understanding of emotions. However, computers are called emotionally challenged because they don't recognize or possess emotions.

To enrich human-computer interface from point-and-click to sense-and-feel, to develop non intrusive sensors, to develop lifelike software agents such as devices, this can express and understand emotion. Today's computer systems have wide range of applications in research areas including security, law enforcement, education and telecommunications. There has been great research on recognizing emotion through facial expressions. There are six basic emotions which are: angry, happy, fear, disgust, sad, surprise. The last is neutral.

Anger is characterized by low and drawn together eyebrows with lines between them, tensed lower lid, bulging or hard stare eyes, firmly pressed together lips, dilated nostrils etc. In happiness, cheeks are raised and teeth exposed with corners of the lips are drawn back and up. Disgust is a particular expression which is very well defined facially with furrowing of eyebrows, closure of eyes and pupil constriction, wrinkled nose, lip retraction and so on.

Through this project, my objective was to primarily only classify between the surprised and the happy faces since they exhibit great degree of similarity and conventionally they've been difficult to differentiate between.

Introduction to Saliency & Saliency Maps

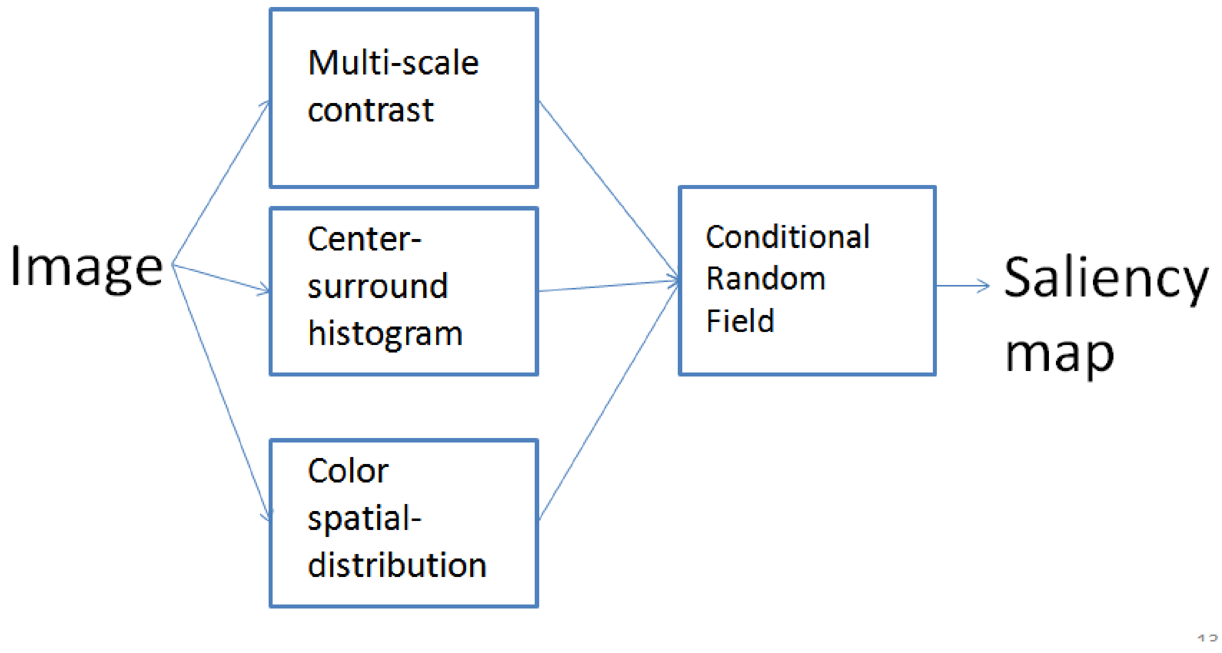
By definition, saliency is the state or quality of an object by virtue of which it stands out from its neighbours. It is a distinct subjective perceptual quality which makes some items attention grabbing because they differ greatly from their neighbours. Some examples of visual salience in their respective contexts are as follows:

- Multiple red coloured balls and a green among them.
- Multiple pens aligned to North-South and one among them aligned East-West.
- Multiple rotating wheels and one among them rotating significantly faster and/or in the reverse direction.
- A lot of vehicles on the highway and one among them is a bicycle and the rest cars.

A saliency map is a topographically arranged map that represents visual saliency of a corresponding visual scene.

Algorithm for constructing saliency maps

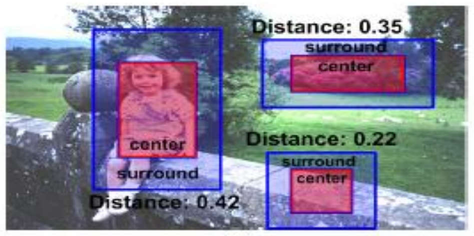
We use the algorithm described by Itti, Koch, Niebur (PAMI 1998) which is a bottom-up approach.



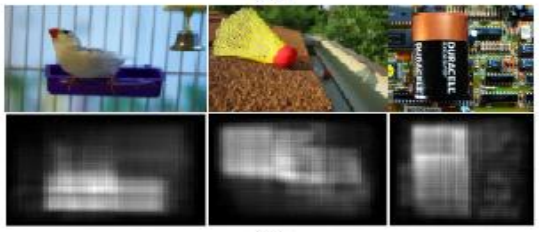
Multi-Scale Contrast:

Local summation of laplacian pyramid

$$f_c(x, I) = \sum_{l=1}^{L} \sum_{x' \in N(x)} \| I^l(x) - I^l(x') \|^2$$



(a)



(b)

Centre-surround histogram:

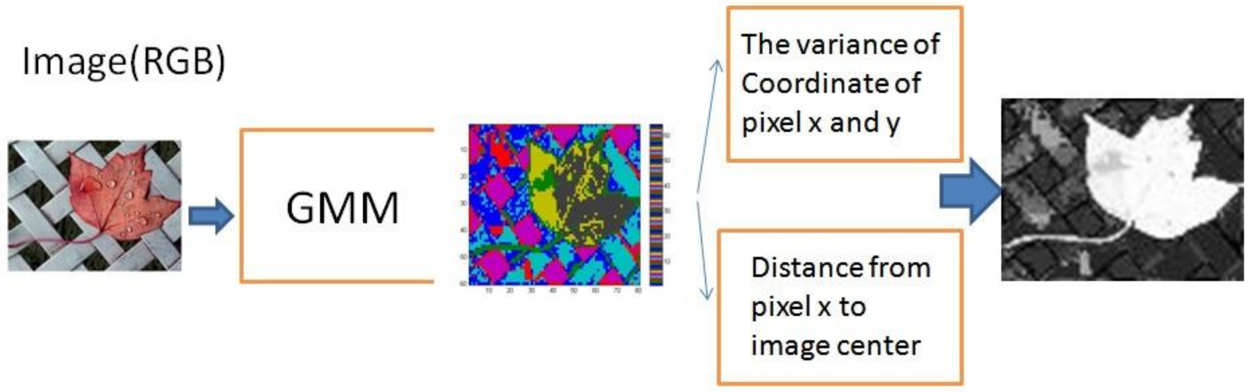
Distance between histograms of RGB color

$$\chi^2(R, R_s) = \frac{1}{2} \sum \frac{(R^i - R_s^i)^2}{(R^i + R_s^i)}$$

$$R^*(x) = \arg\max_{R(x)} \chi^2(R(x), R_s(x))$$

$$f_h(x, I) \propto \sum_{\{x' | x \in R^*(x')\}} \omega_{xx'} \chi^2(R^*(x'), R_s^*(x'))$$

Color spatial-distribution:


Image(RGB)
GMM
The variance of Coordinate of pixel x and y
Distance from pixel x to image center

$$f_s(x,I) \propto \sum_c p(c \mid I_x) \cdot (1 - V(c)) \cdot (1 - D(c))$$

This is what the saliency map algorithm produces as output when run on the same person's happy face and surprised face:

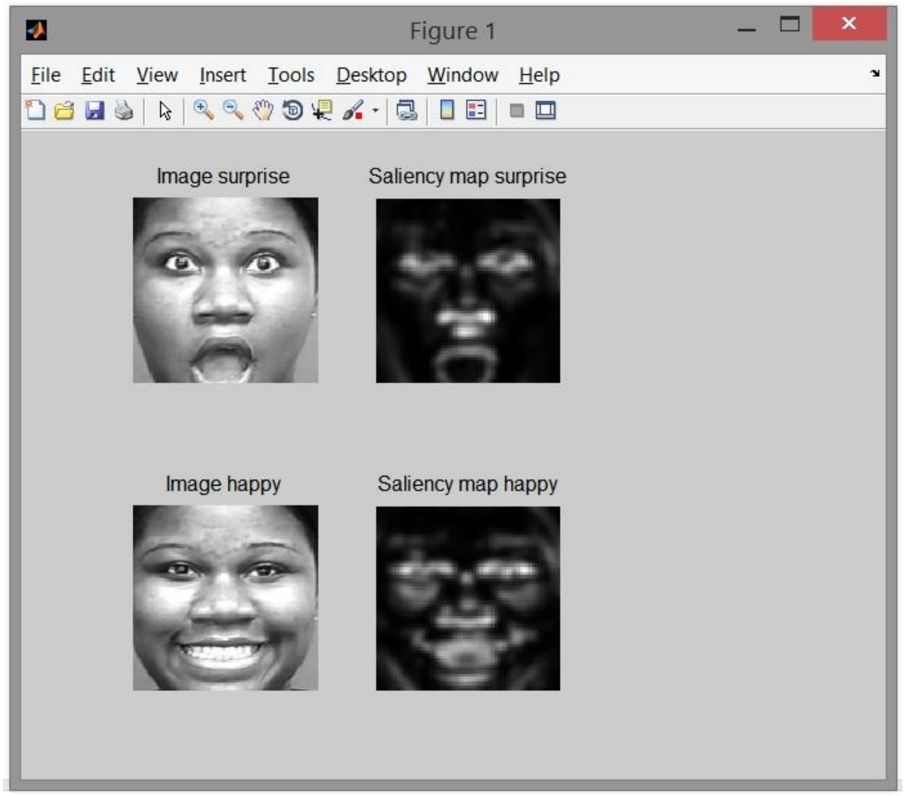
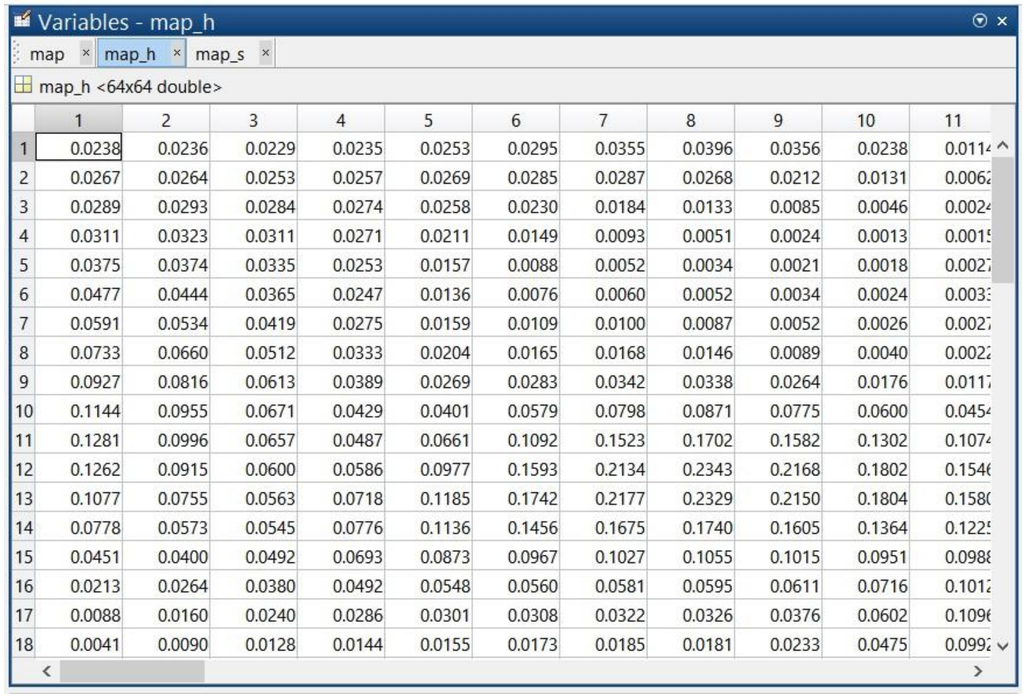


Figure 1. Saliency maps created for happy and surprised expressions of the same model.

The following is an example of the computed values of the map matrix for the woman's happy face:



	1	2	3	4	5	6	7	8	9	10	11
1	0.0238	0.0236	0.0229	0.0235	0.0253	0.0295	0.0355	0.0396	0.0356	0.0238	0.0114
2	0.0267	0.0264	0.0253	0.0257	0.0269	0.0285	0.0287	0.0268	0.0212	0.0131	0.0062
3	0.0289	0.0293	0.0284	0.0274	0.0258	0.0230	0.0184	0.0133	0.0085	0.0046	0.0024
4	0.0311	0.0323	0.0311	0.0271	0.0211	0.0149	0.0093	0.0051	0.0024	0.0013	0.0015
5	0.0375	0.0374	0.0335	0.0253	0.0157	0.0088	0.0052	0.0034	0.0021	0.0018	0.0027
6	0.0477	0.0444	0.0365	0.0247	0.0136	0.0076	0.0060	0.0052	0.0034	0.0024	0.0033
7	0.0591	0.0534	0.0419	0.0275	0.0159	0.0109	0.0100	0.0087	0.0052	0.0026	0.0027
8	0.0733	0.0660	0.0512	0.0333	0.0204	0.0165	0.0168	0.0146	0.0089	0.0040	0.0022
9	0.0927	0.0816	0.0613	0.0389	0.0269	0.0283	0.0342	0.0338	0.0264	0.0176	0.0117
10	0.1144	0.0955	0.0671	0.0429	0.0401	0.0579	0.0798	0.0871	0.0775	0.0600	0.0454
11	0.1281	0.0996	0.0657	0.0487	0.0661	0.1092	0.1523	0.1702	0.1582	0.1302	0.1074
12	0.1262	0.0915	0.0600	0.0586	0.0977	0.1593	0.2134	0.2343	0.2168	0.1802	0.1546
13	0.1077	0.0755	0.0563	0.0718	0.1185	0.1742	0.2177	0.2329	0.2150	0.1804	0.1580
14	0.0778	0.0573	0.0545	0.0776	0.1136	0.1456	0.1675	0.1740	0.1605	0.1364	0.1225
15	0.0451	0.0400	0.0492	0.0693	0.0873	0.0967	0.1027	0.1055	0.1015	0.0951	0.0988
16	0.0213	0.0264	0.0380	0.0492	0.0548	0.0560	0.0581	0.0595	0.0611	0.0716	0.1012
17	0.0088	0.0160	0.0240	0.0286	0.0301	0.0308	0.0322	0.0326	0.0376	0.0602	0.1096
18	0.0041	0.0090	0.0128	0.0144	0.0155	0.0173	0.0185	0.0181	0.0233	0.0475	0.0992

Figure 2. Example of a saliency map data structure created for a happy face

Classification methodology

The features of the dataset were initially the full saliency maps values converted to row vectors. They were then processed to only include the map values of eyes and mouth area. This was done with the objective of feature extraction to reduce computing time, improve accuracy and data dimensionality.

A multi dimensional Support Vector Machine was used as the classification algorithm. In machine learning, a SVM is a discriminative classifier defined by a separating hyper-plane. It can also be used for regression. An SVM model represents the instances as points in space mapped in order to separate the classes by as clear gap as possible. It performs non linear classification by using the kernel trick i.e. implicitly mapping inputs to higher dimensional spaces.

The svm result for the 30 split testing dataset was then compare row by row to the actual label set to create the confusion matrix and count how many instances were correctly predicted.

Dataset information & Pre-processing

The dataset being used initially was the JAFFE (Japanese Female Facial Expression) database which contains 213 images of 7 facial expressions of ten Japanese females. There are 6 basic facial expressions: surprise, happiness, disgust, fear, anger, sadness and a neutral expression. All images were analyzed by sixty Japanese subjects who chose the appropriate emotion for each image. This dataset was abandoned because of less number of samples and because of the characteristic facial structure of Japanese models were not much applicable to the generic situation.

The main dataset used contained the same 7 facial expressions for a more regular set of models. The data was subset into only the happy and surprised expressions and then merged giving a total of 867 instances. This was the standard MMI dataset. The images were then cropped to remove backgrounds and only have minimum necessary facial boundaries.
This data had multiple images of the same shot with minute differences like pupil size changes, slight lighting changes and so on. The classifier was run on this dataset resulting in 100% accuracy. Then, this dataset was reduced to 172 instances after removing such duplicities. The classifier resulted in 100% accuracy on the smaller dataset as well.

Application using MATLAB & Code

Stages:

1. First the entire image dataset names are read.
2. Then we make a saliency map of each image while simultaneously reading it and extracting its correct label from the image name.
3. We extract correct label from image names and associate it to dataset structure, placing it in a separate structure.
4. We make a random permutation of test and training dataset and split the dataset into 30-70 ratios of test and training.
5. We restrict map features to only prominent areas of the image including area around eyes and lips i.e. we do feature extraction.
6. An SVM is trained using the training set and then used to classify the test set. Accuracy and number of correct classifications are observed.
7. The resulting accuracy was 100% for both 172 non repetitive instances of cropped images and 879 repetitive instances.

MATLAB Code for 128 x 128 images:

```
path = 'C:\Users\ABHINAV\Documents\MATLAB\dataset';
dataset = ls('C:\Users\ABHINAV\Documents\MATLAB\dataset');
%dataset = ls(path);
dataset = cellstr(dataset);
dataset = dataset(3:174);
count_dataset = 172;
features=[];
labels=[];
for(i=1:172)
    path = 'C:\Users\ABHINAV\Documents\MATLAB\dataset';
    path = strcat(path, '\');
    path = strcat(path, dataset{i});
    p = default_fast_param;
    p.blurRadius = 0.02;
    img = imread(path);
    map = simpsal(img,p);
    features = [features, map(:)];
    %h = 1, s = 0
    if(isempty(findstr(dataset{i}, 'h.png')))
        value = 0;
    else
        value = 1;
    end
    labels = [labels, value];
end

num_points = size(features, 2);
split_point = round(num_points*0.7);
seq = randperm(num_points);
X_train = features(:,seq(1:split_point)); %features of training set
Y_train = labels(seq(1:split_point)); %labels of training set
X_test = features(:,seq(split_point+1:end)); %features of testing set
Y_test = labels(seq(split_point+1:end)); %labels of testing set

SVMModel = svmtrain(transpose(X_train), transpose(Y_train));
result = svmclassify(SVMModel, transpose(X_test));
count_right = 0;
Y_testt = transpose(Y_test);
for(i=1:numel(Y_testt))
    if(result(i) == Y_testt(i))
        count_right = count_right + 1;
    end
end
```

Saliency Map creation example:

```
%% 1. simplest possible usage : compute standard Itti-Koch Algorithm:

map1 = simpsal('lena.jpg');

%% 2. more complicated usage:

%img = imread('lena.jpg');
p = default_fast_param;
p.blurRadius = 0.02;     % e.g. we can change blur radius
map2 = simpsal(img,p);

subplot(1,3,1);
imshow(img);
title('Original');

subplot(1,3,2);
imshow(map1)
title('Itti Koch');

subplot(1,3,3);
imshow(map2);
title('Itti Koch Simplified');
```

Saliency Map Algorithm code:

```
function [map,chanmaps,maps,chans] = simpsal( img , param );

% outputs:
%
% map : final saliency map
% chanmaps : final saliency map for each channel
% maps : saliency map for each sub-channel, for each channel
% chans : feature maps for each sub-channel, for each channel
%--------------------------------------------------------------

% read in file if img is filename
if ( strcmp(class(img),'char') == 1 ) img = imread(img); end

% convert to double if image is uint8
if ( strcmp(class(img),'uint8') == 1 ) img = double(img)/255; end

% use default parameters if no parameters input
if ( nargin == 1 )
  param = default_pami_param;
end

% compute Gabor filters
if ( isfield(param,'nGaborAngles') && ismember( 'O' , param.channels ) )
  nAngles = param.nGaborAngles;
  maxAngle = 180 - (180/nAngles);
  angs = linspace(0,maxAngle,nAngles);
  param.gabor = {};
  for ang = angs,
    param.gabor{ end + 1 } = simpleGabor( ang );
  end
end

% determine size and number of Center scales
mapSize = [ round(size(img,1) / size(img,2 ) * param.mapWidth) param.mapWidth
];
imgs = {};
if ( param.useMultipleCenterScales == 0 )
  imgs{1} = my_imresize( img , mapSize );
else
  imgs{1} = my_imresize( img , mapSize * 2 );
  imgs{2} = my_imresize( img , mapSize );
  imgs{3} = my_imresize( img , round( mapSize / 2 ) );
end


if (size(img,3)~=3)
  param.channels = setdiff( param.channels , 'CD' );
end

chans = {}; maps = {};
for ci = 1 : numel( param.channels )
  % get feature maps for this channel

  chans{ci} = getchan( imgs , param.channels(ci) , param );

  % compute saliency of each map
  for cj = 1 : length(chans{ci})
    maps{ci}{cj} = zeros( size(chans{ci}{cj}) );
    for fr = 1 : size( chans{ci}{cj} , 3 )
      maps{ci}{cj}(:,:,fr) = pixsal( chans{ci}{cj}(:,:,fr) , param );
    end
  end

  % sum together maps within this channel according to weight
  chanmap = 0;
  sumW = 0;
  for cj = 1 : length(chans{ci})
    if ( param.useNormWeights ) wj = mypeakiness( maps{ci}{cj}(:,:,end) );
    else wj = 1; end

    chanmap = chanmap + wj * imresize(maps{ci}{cj}, mapSize );
    sumW = sumW + wj;
  end
  chanmaps{ci} = chanmap / sumW;
end

% sum together maps across channels
map = 0;
for i = 1 : length( chanmaps )
```

```
  if (param.useNormWeights) wt = mypeakiness( chanmaps{i}(:,:,end) );
  else wt = 1; end
  map = map + wt * chanmaps{i};
end

% apply final blur
if ( param.blurRadius > 0 )
  ker = mygausskernel( param.blurRadius * size(map,1) , 1.5 );
  for fr = 1 : size(map,3)
    map(:,:,fr) = myconv2(myconv2(map(:,:,fr),ker),ker');
  end
end

% apply global center bias
if ( param.centerbias )
  h = size(map,1);
  w = size(map,2);
  map = map .* (gausswin(h,1) * gausswin(w,1)');
end

map = mynorm(map,param);
```

Conclusion

The objective of classifying happy and surprised classes of a facial image expression dataset was completed to a satisfactory accuracy. The saliency map is a very effective and comprehensive way for feature extraction of facial images showing different emotions. An SVM was used to do the classification after extracting features and reducing them. It gave exceptional results.

Further Scope

Android Application

This can be extended towards making an Android application which takes live video feed from the camera and applies the classifier on every nth frame. Most camera phones record 30fps videos and every tenth frame should be used to determine whether the person in the feed is smiling or not. We can also use the 30 fps data as a Time series data and using 2D SVD and a classification algorithm such as Nearest Neighbour, we can do early detection of smiling and even display different stages of smiling.

Extension to 7 class expression recognition

Until now, we have been using only two classes: Happy, Surprise. Instead we can use seven classes, one for each expression. Expressions like happiness, sadness, anger, fear, surprise, and disgust have distinct features and can be given separate classes and the same method of extracting features using Saliency map and classification using any of the classification algorithms can expect fairly good accuracy.

References

[1] Perceptual Organization in Computer Vision: A Review and a Proposal for a Classificatory Structure, Sudeep S. and Kim L. Boyer.

[2] http://www.saliencytoolbox.net/
Dirk Walther and Christof Koch (2006), Modeling attention to salient proto-objects. Neural Networks 19, 1395-1407.

[3] http://www.vision.caltech.edu/~harel/share/gbvs.php
J. Harel, C. Koch, and P. Perona, "Graph-Based Visual Saliency", Proceedings of Neural Information Processing Systems (NIPS), 2006.

[4] J. Harel, A Saliency Implementation in MATLAB:
http://www.klab.caltech.edu/~harel/share/gbvs.php

[5] X. Hou, J. Harel, and C. Koch, "Image Signature: Highlighting Sparse Salient Regions." IEEE Trans. Pattern Anal. Mach. Intell. 34(1): 194-201 (2012)

[6] Itti, L., Koch, C., Niebur, E. (1998) A model of saliency-based visual attention for rapid scene analysis. IEEE Transactions on Pattern Analysis and Machine Intelligence, 20:1254-1259.

[7] Itti, L., Koch, C. (2000) A saliency-based search mechanism for overt and covert shifts of visual attention. Vision Res., 40:1489-1506.

[8] Mai Xu, Yun Ren, Zulin Wang, Learning to Predict Saliency on Face Images.