Automatic facial emotion recognition

Article · January 2005

CITATIONS
45

READS
913

4 authors, including:

Roberto Valenti
University of Amsterdam
32 PUBLICATIONS **1,316** CITATIONS

SEE PROFILE

Automatic facial emotion recognition

Aitor Azcarate, Felix Hageloh, Koen van de Sande, Roberto Valenti

Universiteit van Amsterdam

June 2005

Abstract

Recognizing human facial expression and emotion by computer is an interesting and challenging problem. In this paper we present a system for recognizing emotions through facial expressions displayed in live video streams and video sequences. The system is based on the Piecewise Bézier Volume Deformation tracker [18] and has been extended with a Haar face detector to initially locate the human face automatically. Our experiments with Naive Bayes and the Tree-Augmented-Naive Bayes (TAN) classifiers in person-dependent and person-independent tests on the Cohn-Kanade database [1] show that good classification results can be obtained for facial expression recognition.

1 Introduction

Recently there has been a growing interest in improving the interaction between humans and computers. It is argued that to achieve effective human-computer intelligent interaction, there is a need for the computer to interact naturally with the user, similar to the way humans interact. Humans interact with each other mostly through speech, but also through body gestures to emphasize a certain part of speech and/or display of emotions. Emotions are displayed by visual, vocal and other physiological means. There is more and more evidence appearing that shows that emotional skills are part of what is called 'intelligence' [8]. One of the most important ways for humans to display emotions is through facial expressions. If we want to achieve more effective human-computer interaction, recognizing the emotional state of the human from his or her face could prove to be an invaluable tool.

This work describes a real-time automatic facial expression recognition system using video or webcam input. Our work focuses on initially detecting the human face in the video stream, on classifying the human emotion from facial features and on visualizing the recognition results.

2 Related work

Since the early 1970s there have been extensive studies of human facial expressions. Ekman et al [4] found evidence to support universality in facial expressions. These 'universal facial expressions' are those representing happiness, sadness, anger, fear, surprise and disgust. They studied expressions in many cultures, including preliterate ones, and found much commonality in the expression and recognition of emotions on the face. There are differences as well: Japanese, for example, will suppress their real facial expressions in the presence of the authorities. Babies appear to exhibit a wide range of facial expressions without being taught; this suggests that these expressions are innate [10].

Ekman developed a coding system for facial expressions where movements of the face are described by a set of action units (AUs). Each AU has some related muscular basis. Many researchers were inspired to use image and video processing to automatically track facial features and then use them to categorize

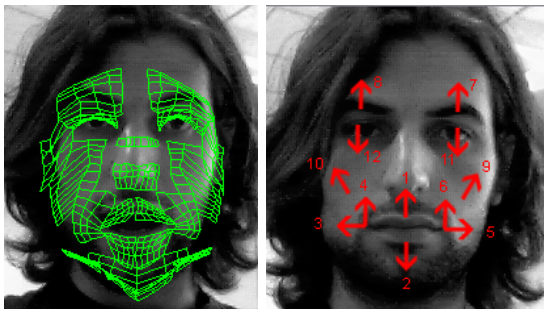

Figure 1: On the left the wireframe model and on the right the facial motion units used in our face tracker.

the different expressions. Pantic and Rothkrantz [13] provide an overview of recent research done in automatic facial expression recognition. Overall the different approaches are similar in that they track facial features using some model of image motion (optical flow, DCT coefficients, etc). Based on the features a classifier is trained. The main difference lies in the set of features extracted from the video images and in the classifier used (often-used classifiers are based on Bayesian approaches or on hidden Markov models). The classifiers used can either be 'static' classifiers or dynamic ones. 'Static' classifiers use feature vectors related to a single frame to perform classification, while dynamic classifiers try to capture the temporal pattern in the sequence of feature vectors related to each frame.

The face tracking we use in our system is based on an incomplete version of the system used in [3]. This system in turn was based on a system developed by Tao and Huang [18] called the Piecewise Bézier Volume Deformation (PBVD) tracker.

This face tracker constructs an explicit 3D wireframe model of the face. In the first frame of the image sequence, landmark facial features such as the eye corners and mouth corners need to be selected by hand. The generic face model consists of 16 surface patches embedded in Bézier volumes and is warped to fit the selected facial features. The surface patches are guaranteed to be continuous and smooth. Once the model is constructed and fitted, head motion and local deformations of the facial features such as the eyebrows, eyelids, and mouth can be tracked. First the 2D image motions are measured using template matching between frames at different resolutions. Image templates from the previous frame and from the very first frame are both used for more robust tracking. The measured 2D image motions are modelled as projections of the true 3D motions onto the image plane. From the 2D motions of several points on the mesh, the 3D motion can be estimated. Figure 1 shows an example of one frame with the wireframe model overlayed on the face being tracked. The recovered motions are represented in terms of magnitudes of some predefined motion of various facial features. Each feature motion corresponds to a simple deformation on the face, defined in terms of the Bézier volume control parameters. We refer to these motions vectors as Motion-Units (MU's). Note that they are similar but not equivalent to the AUs of Ekman. The MU's used in the face tracker are shown in figure 1 on the right and are described in Table 1. These MU's are the features we use as input to our classifiers described in later sections.

AU	Description
1	vertical movement of the center of upper lip
2	vertical movement of the center of lower lip
3	horizontal movement of left mouth corner
4	vertical movement of left mouth corner
5	horizontal movement of right mouth corner
6	vertical movement of right mouth corner
7	vertical movement of right brow
8	vertical movement of left brow
9	lifting of right cheek
10	lifting of left cheek
11	blinking of right eye
12	blinking of left eye

Table 1: Motion units used in our face tracker.

3 Classifiers

Naive Bayes classifiers are popular due to their simplicity and their success in past applications. The

simplicity of a naive Bayes classifier stems from its independence assumption, which assumes that features are uncorrelated. Thus their joint probability can be expressed as a product of their individual probabilities. As in any classification problem we would like to assign a class label c to an observed feature vector $\mathbf{X}$ with n dimensions (features). The optimal classification rule under the maximum likelihood (ML) framework to classify an observed feature vector of n dimensions, $\mathbf{X} \in R^n$, to one of $|C|$ class labels, $c \in \{1, ..., |C|\}$, is given as:

$$\hat{c} = argmax_c P(X|c; \Theta). \quad (1)$$

where Θ is the set of parameters that need to be learned for the classifier. Given the naive Bayes assumption, the conditional probability of $\mathbf{X}$ given a class label c is defined as:

$$P(X|c; \Theta) = \prod_{i=1}^{n} P(x_i|c; \Theta). \quad (2)$$

Having a continuous feature space - which is true in our case - the conditional probabilities for each feature can be modelled as probability distribution functions. The Gaussian distribution is most commonly used and ML methods are used to estimate its parameters. For a naive Bayes classifier we have to learn a distribution for each feature, but since we are dealing with only one dimension, the parameters for the Gaussian distribution (mean and variance) can easily be calculated.

However, assuming Gaussian distributions is not always accurate and thus the Cauchy distribution was proposed as an alternative by Sebe et al [17]. While it can give better classification results in some cases, its main drawback is that its parameters are much more difficult to estimate.

Despite the seemingly weak independence assumption of the naive Bayes classifier, it normally gives surprisingly good results. Recent studies [5, 7] also give some theoretical explanation for this success. Nevertheless, in cases were there are dependencies among features, the naive Bayes model certainly gives a sub-optimal solution. In our scenario it is feasible to assume some dependence between features due to the anatomic structure of the face. Hence we should attempt to also find these dependencies and model their joint distributions. Bayesian networks are an intuitive and efficient way to model such joint distributions, and they are also suitable for classification. In fact, the naive Bayes model is actually an extreme case of a Bayesian network where all nodes are only connected to the class node (i.e. there are no dependencies between features modelled).

A Bayesian network consists of a directed acyclic graph in which every node is associated with a variable X_i and with a conditional distribution $P(X_i|\Pi_i)$, where Π_i denotes the parents of X_i in the graph. The joint probability distribution is then defined as:

$$P(X_1, \ldots, X_n) = \prod_{i=1}^{n} P(X_i|\Pi_i)$$

One of the important aspects when designing a Bayesian network classifier is choosing the right structure for the network graph. Choosing a wrong structure can have dire effects on the classification results. When the structure of the Bayesian network is unknown or uncertain, as it is the case here, it is better to learn the optimal structure using ML. However, this requires searching through all possible structures, i.e. all possible dependencies among features, which is a NP-complete problem. Thus we should restrict ourselves to a smaller class of structures to make the problem tractable. One such class of structures was proposed by Friedman et al [6] and is referred to as the Tree-Augmented-Naive Bayes (TAN) classifier. TAN classifiers have the advantage that there exists an efficient algorithm [2] to compute the optimal TAN model.

TAN classifiers are a subclass of Bayesian network classifiers where the class node has no parents and each feature has a parent the class node and at most one other feature. To learn its exact structure, a modified Chow-Liu algorithm [2] for constructing tree augmented Bayesian networks [6] is used.

Essentially the algorithm builds a maximum weighted spanning tree between the feature nodes. As weights of the arcs the pairwise class-conditional mutual information among the features is used. The resultant graph of the algorithm is a tree including all feature pairs that maximizes the sum of the weights

of the arcs. To make the undirected tree a directed graph, a root node is chosen and all edges are made to point away from the root node. Then the class node is made parent node of all features to construct the final TAN. The detailed algorithm and the algorithm used to compute the maximum spanning tree can be found in [3].

The last step is to compute the joint distributions of the nodes. Again Gaussian distributions are used and estimated using ML techniques. This is essentially the same as for the naive Bayes classifier, only that now we need to compute additional covariance parameters.

Our project aims to design a dynamic classifier for facial expressions, which means also taking temporal patterns into account. To classify an emotion not only the current video frame is used, but also past video frames. While [3] proposes a multi-level Hidden Markov Model based classifier, the current implementation only takes temporal patterns into account by averaging classification results over a set number of past frames. We do not discuss dynamic classifiers and the proposed Hidden Markov Model further, because we did not work on extending the system in this direction.

4 Face detection

As we described in section 2, the existing system required placing all marker points on landmark facial features manually. To automate this, we want to detect the initial location of the human face automatically and use this information to place the marker points near their landmark features. We do this by placing a scaled version of the landmark model of the face on the detected face location.

As our face detector, we chose a fast and robust classifier proposed by Viola and Jones [19] and improved by Lienhart et al [11, 12]. Their algorithm makes three main contributions:

- The use of integral images.
- A selection of features through a boosting algorithm (Adaboost)

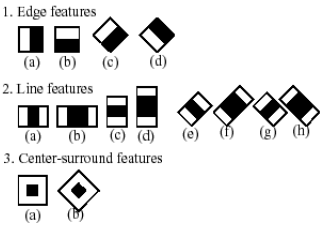

Figure 2: Haar features.

- A method to combine simple classifiers in a cascade structure

4.1 Integral Images

Analyzing images is not an easy task. Using just the pixel information can be useful in some fields (i.e. movement detection) but is in general not enough to recognize a known object. In 1998, Papageorgiou et al [14] proposed a method to analyze image features using a subgroup of Haar-like features, derived from the Haar transforms. This subgroup was extended later by Lienhart et al [11] to also detect small rotations of the sought-after object. The basic classifiers are decision-tree classifiers with at least 2 leaves. Haar-like features are the input to the basic classifiers and are calculated as described below. The algorithm we are describing uses the Haar-like features shown in figure 2.

The feature used in a particular classifier is specified by its shape (1a, 2b, etc), position within the region of interest and the scale (this scale is not the same as the scale used at the detection stage, though these two scales are multiplied). For example, in case of the third line feature (2c) the response is calculated as the difference between the sum of image pixels under the rectangle covering the whole feature (including the two white stripes and the black stripe in the middle) and the sum of the image pixels under the black stripe multiplied by 3 in order to compensate for the differences in the size of areas. Calculating sums of pixels over rectangular regions can be very expensive in computational terms, but this problem can be solved by using an intermediate representation

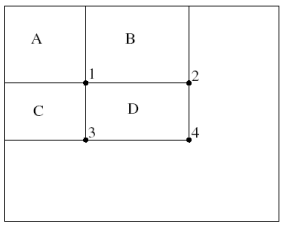

Figure 3: Calculation of the rectangular regions.

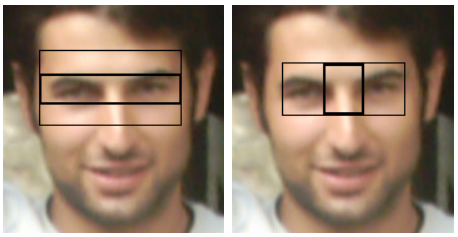

Figure 4: First two iterations of Adaboost.

of the images, namely integral images.

Those intermediate images are easily generated by the cumulative sums of the original image's pixels: every pixel of the integral image $ii(x, y)$ corresponds to the sum of all the pixels in the original image i from $i(0, 0)$ to $i(x', y')$.

$$ii(x, y) = \sum_{x' \leq x, y' \leq y} i(x', y')$$

Using recursive formulas, it is possible to generate an integral image from an original with a single computational step:

$$s(x, y) = s(x, y - 1) + i(x, y)$$

$$ii(x, y) = ii(x - 1, y) + s(x, y)$$

where $s(x, y)$ is the cumulative sum of the row.

Once an integral image is generated, it is rather easy to calculate the sum of pixels under an arbitrary rectangular region D using the values of points 1, 2, 3 and 4. This is illustrated in figure 3.

In fact, the value of point 1 is the cumulative sum of A, point 2 is the cumulative sum of $A + B$, point 3 is $A + C$ and point 4 is $A + B + C + D$. Since we are looking for the value of D, we should subtract from the value of point 4 the value of point 3 and the value of point 2, and add the value of point 1 since it was subtracted twice during the previous operation.

4.2 Feature selection using Adaboost

Proposed by Schapire [15, 16], the Adaboost algorithm is used to 'boost' the performance of a learning algorithm. In this case, the algorithm is used both to train the classifiers and to analyze the input image. In a 24x24 pixel image, there are over 180.000 Haar-like features that can be detected, a lot more than the number of pixel in the image (576). In case we are dealing with a bigger image, the number should be multiplied for all the sub-windows of 24 pixels in the image. The computational cost of this operation is clearly prohibitive. Instead, Adaboost is used to select which of the features are actually relevant for the sought-after object, drastically reducing the number of features to be analyzed. In every iteration, Adaboost chooses the most characterizing feature in the entire training set from the 180.000 features possible in every image.

The first two selected feature are displayed in figure 4: it is clear that the most discriminative feature is the difference between the line of the eyes and the surrounding; for a face the surroundings are lighter than the eyes themselves. The second feature selected is the difference in tonality between the eyes and the nose; the nose is also lighter when compared to the area of the eyes. The algorithm will continue to select good features that can be combined in a classifier.

4.3 Cascade of classifiers

Every step, a simple classifier (also called weak because of their low discriminative power) is built. The combination of all the weak classifiers will form a strong classifier that can recognize any kind of object it was trained with. The problem is to search for this particular sized window over the full picture, applying the sequence of weak classifiers on every sub-window of the picture. Viola and Jones [19] used a cascade of classifiers (see figure 5) to tackle

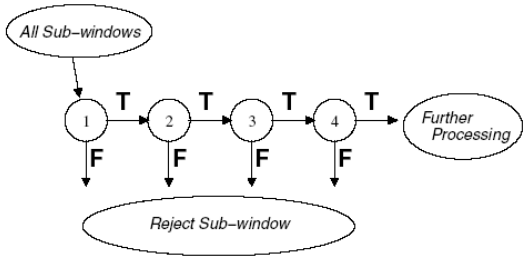

Figure 5: Cascade of classifiers.

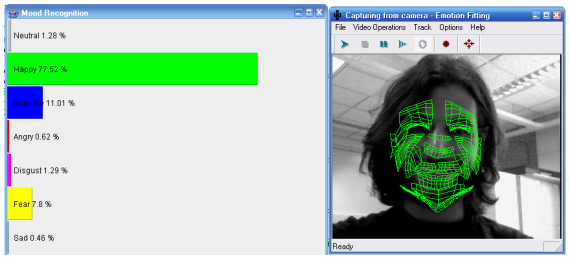

Figure 6: Bars visualization of the probabilities for each emotion.

this problem: the first classifier (the most discriminative) is applied to all the sub-windows of the image, and at different scale. The second classifier will be applied only to the sub-windows in which the first classifier succeeded. The cascade continues, applying all the weak classifiers and discarding the negative sub-windows, concentrating the computational power only on the promising areas.

5 Implementation

When studying the incomplete existing implementation we received, we decided to remove the outdated parts and to change the program structure to be able to create a distributable package, executable by a normal user without Visual C++ and the required libraries installed. Minor code cleaning and bugfixing was performed all over the source code.

Another big change was in the source of the input videos, which supported only AVI movies and Matrox cameras. We implemented a new class based on the OpenCV library [9] which uses the same code to read from any kind of movie file and virtually all cameras supporting computer attachment. It is now possible to select the camera's options directly and record the video stream directly from the emotion fitting program. On the interface, new buttons were added to control the new options, while the old ones were debugged and restyled in a modern look.

As stated in the introduction, our main contribution is the inclusion of a face detector through the OpenCV library: we used it to snap the position and the scale of the markers to the position and scale of the user's face, and most importantly to reinitialize the position of the mesh when the face was lost during the emotion fitting. This contribution made the program more usable and robust, introducing brief errors only in some cases of occlusion or fast movements of the user. Furthermore, the communication between the video program and the classifier program was reimplemented to reduce the delay that were previously introduced by establishing a new connection for every image frame.

5.1 Visualization

For the visualization of the emotions we chose two different forms. The first uses the sizes of bars to display the emotion and the second uses a circle. Every emotion has a different color. For example happy has the color green due to the fact that green is generally considered a 'positive color' and angry has the color red because red is generally considered a 'negative color'. For clarity we also write the emotion and corresponding probability percentage in the mood window. The mood with highest probability is also written separately. In the mood window there are two combo boxes at the bottom. In these combo boxes there is the possibility of choosing the visualization type and the classifier. Figure 6 shows the bars visualization. If the program is 100% sure that we have a certain emotion, then the width of the bar will correspond to the full width of the window.

Figure 7 shows the circle visualization. The edge

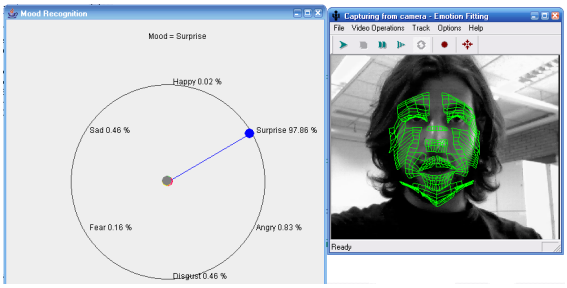

Figure 7: Circle visualization of the probabilities for each emotion

of the circle is a classification of 100% of the emotion. So if the dots get closer to the edge the higher the probability of the emotion. The center of the circle corresponds to neutral. The current mood is displayed on the top of the window.

6 Evaluation

We ran several test to evaluate the performance of the emotion detector. Note that our changes, fixes and new implementation of classifiers should not alter the previously reported results [3]. The aim of our experiments is thus getting a second set of results for comparison purposes.

6.1 Dataset

Our dataset is the Cohn-Kanade database [1], which contains 52 different people expressing 7 emotions. These emotions are: neutral, happy, surprised, angry, disgusted, afraid and sad. For every person several videos are available. Every video starts with the neutral expression and then shows an emotion. Each frame of every video is labelled with the corresponding emotion. For some people in the database, not all emotions are available.

6.2 Experiments

For each classifier we performed person dependent and person independent test. The training set for person independent tests contains samples from several people displaying all seven emotions. A sample consists of a single labelled frame from a video. The test set is a disjoint set with samples from other people. On the other hand, in person dependent tests the training set contains samples from just a single person. It is then evaluated on a disjoint test set containing only samples from the same person.

6.3 Results

First we examined the performance of our implementation of a Naive Bayes classifier. We divided the data into three equal parts, from which we used two parts for training and one part for testing. Results are averaged over the three different combinations of test/training set possible. This is also known as cross-validation. The confusion matrix of the person independent test is shown in table 2. The confusion matrix for the TAN classifier, using the same training and test sets, is shown in table 3.

In person dependent tests the classifier is trained and evaluated using data from only a single person. All samples for a person are again split in three equal parts for cross-validation. We did this for five people and averaged the results to obtain the confusion matrix. The confusion matrix of the person dependent test is shown in table 4. The confusion matrix for the TAN classifier using the same people is shown in table 5.

As can be seen in the confusion matrices the results of classifying the emotion in the person dependent tests are better (for NB 64,3% compared to 93,2% and for TAN 53,8% compared to 62,1%) than the person independent tests. This result is of course intuitively correct, because the classifier was trained specifically for that person, so it should perform quite well when the test set is also from that same person.

Our results very clearly do not correspond to previously reported results by Cohen et al [3]. Surprisingly our Naive Bayes classifier outperforms the TAN classifier. Our Naive Bayes classifier gives the same results as reported in literature. The TAN classifier, however, performed significantly worse. We presume this is caused by an incorrectly learned dependency structure for the TAN model. Investigat-

	Neutral	Happy	Surprised	Angry	Disgusted	Afraid	Sad
Neutral	**82.34**	1.89	1.76	1.78	0.89	3.74	7.60
Happy	2.17	**74.17**	0.42	1.95	3.81	14.85	2.63
Surprised	2.16	0.00	**90.08**	1.35	0.00	1.60	4.81
Angry	8.01	5.43	0.31	**55.28**	20.96	3.60	6.42
Disgusted	6.12	8.66	3.76	23.76	**46.54**	6.93	4.24
Afraid	4.15	20.52	12.91	0.08	1.66	**57.47**	3.22
Sad	22.46	2.82	15.26	7.95	6.17	1.38	**43.96**

Table 2: Confusion matrix for the naive Bayes classifier in person independent tests. The rows represent the emotion expressed and the columns represent the emotion classified. Average accuracy is 64.3%. Rows represent the true emotion, while columns represent the detected emotion.

	Neutral	Happy	Surprised	Angry	Disgusted	Afraid	Sad
Neutral	**87.35**	1.49	1.66	2.51	0.37	2.58	4.04
Happy	6.63	**63.98**	2.04	2.42	5.31	14.05	5.57
Surprised	3.90	0.00	**80.97**	1.82	0.74	2.29	10.28
Angry	17.93	6.43	4.25	**36.32**	15.72	9.94	9.40
Disgusted	9.33	9.18	4.11	25.45	**37.07**	7.68	7.19
Afraid	11.76	22.47	10.92	4.89	5.75	**37.08**	7.13
Sad	21.14	9.10	11.24	9.09	5.71	9.82	**33.90**

Table 3: Confusion matrix for the naive TAN classifier in person independent tests. Average accuracy is 53.8%.

ing the learned dependencies, we found them to disagree greatly with the ones reported by Cohen et al. While they reported mostly horizontal dependencies between the features on the face, our structure contains many vertical dependencies. This could be a bug in our implementation of the TAN classifier.

Another possible explanation is that the TAN classifier lacks enough training data to be effectively trained. This often happens with more complex classifiers, because they need to estimate more classifier parameters from the same amount of data.

Looking for patterns in the confusion matrices, we see that the 'positive' emotions happy and surprised are recognized very well; these are very pronounced emotions. It holds for all emotions that when they are not pronounced enough, they can be misclassified as neutral instead of the correct emotion. Happy is confused most often with afraid, and the converse also holds. Analysis shows that people who are afraid tend to open their mouth a bit and the mouth corners are up a bit. When looking at just a single frame, it is very hard to distinguish these two emotions. We can make a similar point for anger and disgust: both curve the mouth downward, though people tend to open their mouth a bit with disgust and close it when they are angry.

An interesting emotion is fear (afraid), as it can be misclassified as surprise quite often, while the converse seldomly happens. We think that these emotions are very similar in their expression (e.g. 'close' to each other) but that surprise has a very specific expression (little variation in the expression), making it easy to recognize. Fear, however, probably has a range of forms it can take and we think that surprise may be positioned in-between these forms. The main confusion for fear is happiness; again in this confusion the mouth movement is similar, but for these two emotions also the eyebrows also tend to be raised

	Neutral	Happy	Surprised	Angry	Disgusted	Afraid	Sad
Neutral	**88.17**	2.62	1.83	1.47	2.29	0.56	3.07
Happy	2.22	**95.16**	0.00	0.00	0.00	2.62	0.00
Surprised	0.00	0.00	**100.00**	0.00	0.00	0.00	0.00
Angry	1.67	0.00	0.00	**98.33**	0.00	0.00	0.00
Disgusted	10.00	2.22	0.00	4.44	**81.11**	0.00	2.22
Afraid	3.56	0.00	0.00	0.00	0.00	**94.22**	2.22
Sad	4.44	0.00	0.00	0.00	0.00	0.00	**95.56**

Table 4: Confusion matrix for the naive Bayes classifier in person dependent tests. Average accuracy is 93.2%. Rows represent the true emotion, while columns represent the detected emotion. Results averaged over 5 people.

	Neutral	Happy	Surprised	Angry	Disgusted	Afraid	Sad
Neutral	**95.26**	0.42	0.39	2.09	0.00	0.00	1.84
Happy	20.56	**56.98**	2.50	11.35	0.00	5.28	3.33
Surprised	12.62	1.11	**73.60**	8.78	0.00	2.22	1.67
Angry	15.78	2.78	0.00	**79.22**	0.00	0.00	2.22
Disgusted	27.78	7.78	2.22	18.89	**33.33**	2.22	7.78
Afraid	30.22	11.00	0.00	9.33	2.22	**41.67**	5.56
Sad	35.11	0.00	4.44	4.44	1.33	0.00	**54.67**

Table 5: Confusion matrix for the naive TAN classifier in person independent tests. Average accuracy is 62.1%. Results averaged over 5 people.

a bit. Discriminating these two emotions manually from a single frame ourselves is hard, so this makes sense.

7 Conclusion

We significantly improved the usability and user-friendliness of the existing facial tracker, extending it with automatic face positioning, emotion classifiers and visualization. Our Naive Bayes emotion classifier performs quite well. The performance of our TAN classifier is not up to par with existing research. The classifier either lacks enough training data, or has an implementation problem.

We believe that additional improvements to the system are possible. First of all we could use specialized classifiers to detect specific emotions, and combine them to improve the classification performance. Furthermore, the current classifier shows a strange behavior when readapting the mask after it loses it, due a continuous classification of the deformations. Those deformations are artificial and generated during the re-adaptation step and should not be considered for classification, so classification should be interrupted during mesh repositioning. Another important step is to make the system more robust to lighting conditions and partial occlusions. In fact, the face detector will work only if all the features from the face are visible and won't work if the face is partially occluded or not in a good lighting condition. Finally, the system should be more person independent: with the current implementation, the system requires markers to let the user select the important feature of the face. This should be transparent to the user, using the face detector to localize the position and the scale of the face and sequentially apply another algorithm to adjust those markers to

the current face. In this way, there will be no need for markers anymore and the system could be used by any user, without any intervention. With these improvements, this application could be applied to real-life applications such as games, chat programs, virtual avatars, interactive TV and other new forms of human-computer interaction.

References

[1] J. Cohn, T. Kanade Cohn-Kanade AU-Coded Facial Expression Database *Carnegie Mellon University*

[2] C.K. Chow, C.N. Liu. Approximating discrete probability distributions with dependence trees. *IEEE Trans. Information Theory*, 14:462–467, 1968.

[3] I. Cohen, N. Sebe, A. Garg, L. Chen, and T.S. Huang. Facial expression recognition from video sequences: Temporal and static modeling. *Computer Vision and Image Understanding*, 91(1-2):160–187, 2003.

[4] P. Ekman Strong evidence for universals in facial expressions. *Psychol. Bull.*, 115(2): 268–287, 1994.

[5] J.H. Friedman On bias, variance 0/1-loss, and the curse-of-dimensionality. *Data Mining Knowledge Discovery*, 1 (1): 55–77, 1997.

[6] N. Friedman, D. Geiger, M. Goldszmidt. Bayesian network classifiers. *Machine Learning*, 29(2):131–163, 1997.

[7] A. Garg, D. Roth. Understanding probabilistic classifiers. *Proc. Eur. Conf. on Machine Learning*, 179–191, 2001.

[8] D. Goleman. Emotional Intelligence. *Bantam Books*, New York, 1995.

[9] Intel Research Laboratories. OpenCV: Open computer vision library. `http://sf.net/projects/opencvlibrary/`.

[10] C.E. Izard. Innate and universal facial expressions: evidence from developmental and cross-cultural research. *Psychol. Bull.*, 115(2): 288–299, 1994.

[11] R. Lienhart, J. Maydt. An extended set of haar-like features for rapid object detection. *Proceedings of the IEEE International Conference on Image Processing*, Rochester, New York, vol. 1, pp. 900-903, 2002.

[12] R. Lienhart, A. Kuranov, V. Pisarevsky. Empirical Analysis of Detection Cascades of Boosted Classifiers for Rapid Object Detection. *Intel Corporation*, Technical report, 297–304, 2002.

[13] M. Pantic, L.J.M. Rothkrantz. Automatic analysis of facial expressions: the state of the art. *IEEE Trans. PAMI*, 22(12): 1424–1445, 2000.

[14] C. Papageorgiu, M. Oren, T. Poggio. A general framework for Object Detection. *Proceedings of the International Conference on Computer Vision*, Bombay, India, pp. 555-562, 1998.

[15] R. Schapire, Y. Freund. Experiments with a new boosting algorithm. *Proceedings of the International Conference on Machine Learning*, Bari, Italy, Morgan Kaufmann, pp. 148-156, 1996.

[16] R. Schapire. The strenght of weak learnability. *Machine Learning*, 5(1), 197-227, 1990.

[17] N. Sebe, I. Cohen, A. Garg, M.S. Lew, T.S. Huang. Emotion Recognition Using a Cauchy Naive Bayes Classifier. *International Conference on Pattern Recognition (ICPR02)*, vol I, pp. 17–20, Quebec, Canada, 2002.

[18] H. Tao, T.S. Huang. Connected vibrations: a modal analysis approach to non-rigid motion tracking. *Proc. IEEE Conf. on CVPR*, 735–740, 1998.

[19] P. Viola, M. Jones. Rapid Object Detection Using a Boosted Cascade of Simple Features. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Kauai, Hawaii, vol. 1, pp. 511-518, 2001.