Expressive MPEG-4 Facial Animation Using Quadratic Deformation Models

Mohammad Obaid[*†], Ramakrishnan Mukundan[†], Mark Billinghurst[*], and Catherine Pelachaud[‡]

[*]HITLab New Zealand, University of Canterbury, Christchurch, New Zealand

[†]Computer Science and Software Engineering Dept., University of Canterbury, New Zealand

[‡]CNRS - Telecom ParisTech, Paris, France

Email: {mohammad.obaid, mark.billinghurst}@hitlabnz.org,

mukund@cosc.canterbury.ac.nz, catherine.pelachaud@telecom-paristech.fr

Abstract—In this paper we propose an approach compliant with the MPEG-4 standard to synthesize and control facial expressions generated using 3D facial models. This is achieved by establishing the MPEG-4 facial animation standard conformity with the quadratic deformation model representations of facial expressions. This conformity allows us to utilize the MPEG-4 facial animation parameters (FAPs) with the quadratic deformation tables, as a higher layer, to compute the FAP values.

The FAP values for an expression E are computed by performing a linear mapping between a set of transformed MPEG-4 FAP points (using quadratic deformation models) and the 3D facial model semantics. The nature of the quadratic deformation model representations of facial expressions can be employed to synthesize and control the six main expressions (smile, sad, fear, surprise, anger, and disgust). Using Whissel's psychological studies on emotions we compute an interpolation parameter that is used to synthesize intermediate facial expressions.

The paper presents results of experimental studies performed using the Greta embodied conversational agent. The achieved results are promising and can lead to future research in synthesizing a wider range of facial expressions.

Index Terms—MPEG-4 Facial Animation, Quadratic Deformation Models, Facial Animation, Facial Expression synthesize.

I. INTRODUCTION

Expressive 3D virtual characters are an emerging technology metaphor among several sectors such as human computer interaction, games and e-learning. Introducing them in computerized systems simulates, as possible, the seamless natural interaction between the user and the computer.

Recently, the importance of emotional expression in 3D characters has been recognized and discussed. Expressive 3D characters can have several possible channels of expression such as gestures, facial expressions, gaze or posture [1]. Facial expressions are described to be the most expressive communicative channel [1][2]. One of the main challenges in synthesizing facial expressions is producing techniques to generate and control facial animations on a 3D virtual character compliant with the MPEG-4 facial animation standard.

In this paper we propose a mechanism to synthesize facial expressions for 3D characters by combining the MPEG-4 facial animation standard with the quadratic deformation model representation of facial expressions [3]. To illustrate our proposed approach we use the Embodied Conversational Agent (ECA) Greta [4] to generate different expressive states of the agent.

The rest of the paper is organised as follows. Section II gives an review of the related work and background in the area of facial expression representations. Section III illustrates an overview of the MEPG-4 facial animation using quadratic deformation models. Section IV presents the quadratic deformation model representations of facial expressions. Section V explains the quadratic deformation model conformity with the MPEG-4 facial animation standard. Section VI gives a description of how the MPEG-4 FAPs are computed using the quadratic models. Section VII describes a technique to generate different facial expressions. Finally, we show the results of our approach and conclude the paper respectively.

II. RELATED WORK

In the past three decades, research and development in facial animation has advanced to include several standards that define how to perform facial expressions in synthetic 3D virtual characters. Paul Ekman and Wallace Friesen [5] have developed the most significant description of facial expressions; the Facial Action Coding System (FACS). FACS defines facial expressions as a combination of facial actions corresponding to particular muscle group movements. Ekman and Friesen defined these as Action Units (AU).

In general many researchers have based their work on the FACS system, which is to this date the most commonly used standard. However, animators generally find the FACS system to be a very abstract standard, as it gives a general description of muscle movements. The MPEG-4 facial animation standard [6] was developed as a successor to FACS to be the first parameterization standard for facial animations. This standard supports the definition, encoding, and transition of facial animation. The facial features specified by the MPEG-4 standard are coded by 68 Facial Animation Parameters (FAP) [6].

One of the limitations of the MPEG-4 facial animation standard is that it does not support a holistic definition and manipulation mechanism for facial expressions and can be considered a low level parameterization standard. In general an upper layer definition is required on the top of the MPEG-4 standard, to allow for the ease of synthesizing, controlling and manipulating facial animations using FAPs.

In recent years, several facial animation techniques have been proposed to synthesize and generate facial expressions for 3D virtual characters such as the work found in [7] and [8]. Moreover, since the establishment of the MPEG-4 facial animation standard, several 3D synthetic virtual characters and ECAs have emerged to support the MPEG-4 standard. One of the most well known engines developed based on the MPEG-4
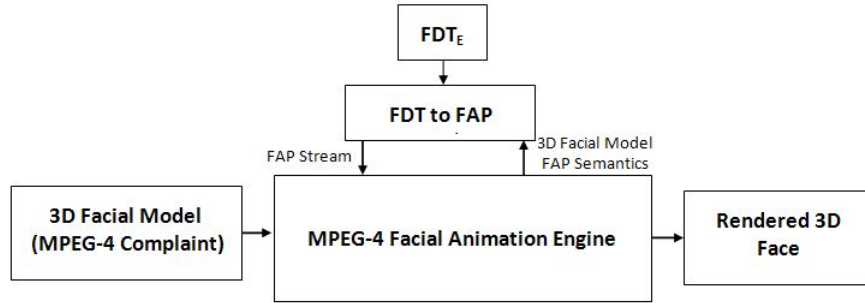
Fig. 1. FDT to FAP mapping as part of the MPEG-4 facial animation engine.

standard is the Greta ECA engine [4]. The design of Greta's face is supported by a 3D facial model compliant with MPEG-4 facial animation standard and it is capable of simulating, in a believable manner, the dynamic aspects of the human face.

Several approaches have been proposed to synthesize and animate facial expressions based on MPEG-4, this includes the work by Paradiso [9] who proposed a method to define a set of algebra operators for combining facial animations and expressions based on MPEG-4 FAPs. Raouzaiou et al. [10] developed a framework to synthesize and analyze intermediate and primary expressions by creating visual profiles based on MPEG-4 FAPs. Their main focus was to synthesize and animate facial expressions on a 3D facial model compliant with the MPEG-4 standard.

More recently, Obaid et al. [3] represented facial expressions based on quadratic deformation models applied to muscle regions. In their approach they construct Facial Deformation Tables (FDTs) for each of the main universal expressions, by capturing the facial muscle deformation using the most general rubber-sheet transformation of second degree. In this paper, we propose an approach to synthesize facial expressions by utilizing the MPEG-4 facial animation parameters with quadratic deformation model representations of facial expressions. The novelty of our approach is being able to synthesize and control facial expressions using real muscle quadratic deformation parameters of facial expressions. The proposed approach allows us to synthesize not only the primary expression but also intermediate expressions.

III. OVERVIEW OF MPEG-4 FACIAL ANIMATION USING FDT

In general, most MPEG-4 compliant facial models require a facial animation engine to derive the final rendering of the 3D facial appearance. The facial animation process starts by providing the facial animation engine with a 3D facial model geometry and its semantic rules (such as FAPs). The engine then automatically computes the animation rules based on a stream of FAP values coming from a higher layer input. Finally, the 3D face is rendered based on the computed animation rules. The approach described in this paper acts as the higher layer input that defines and computes the stream of FAP values based on the 3D model semantic rules and

using the FDT parameters. The general picture of the process is shown in Fig. 1.

The aim of our approach is to synthesize and control facial expressions using real muscle deformations of facial expressions (described by Obaid et al. [3]). To achieve this, we first check that the conformity of the quadratic deformation models with the MPEG-4 facial animation standard (as described in section V). We then describe a mechanism to compute the mapping between the quadratic deformation models and FAPs (as described in section VI). Finally, we describe an interpolation technique to synthesize and control facial expressions.

In the following sections we describe how the quadratic deformation model representations of facial expressions, can be used to synthesize and control the facial expression of a 3D facial model compliant with MPEG-4 facial animation standard.

IV. FACIAL EXPRESSIONS USING QUADRATIC DEFORMATION MODELS

Obaid et al. [3] represented facial expressions by capturing the non-linear nature of muscle deformations for each expression. The following three steps are used in capturing the muscle deformations:

1) Subdivide the face into 16 muscle based facial regions, as shown in Fig. 2.
2) Use the most general second degree rubber-sheet transformation to derive the deformation parameters for each region by applying the least-square minimization technique.
3) Construct Facial Deformation Tables (FDT) to mathematically represent each expression. The procedure to compute the FDT_E for each of the six main expressions E (smile, sad, fear, surprise, anger and disgust) is described in [3].

From Obaid et al. [3], rubber-sheet transformations are higher-order (non-linear) polynomial transformations [11]. The name comes from the logical analogy of overlaying an elastic piece of rubber to fit over a surface of some shape. In the two-dimensional space, rubber-sheet transformations are defined by a pair of equations:
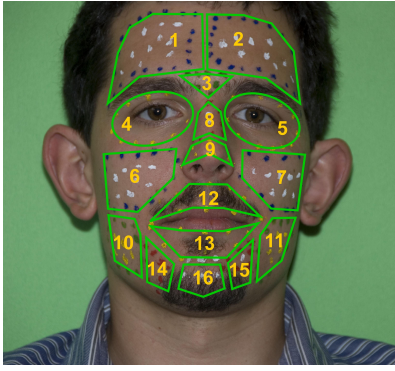
Fig. 2. The sixteen Facial Muscle Deformation Regions described by [3].

$$x_i' = a_1 x_i^2 + a_2 x_i y_i + a_3 y_i^2 + a_4 x_i + a_5 y_i + a_6 \qquad (1)$$
$$y_i' = b_1 x_i^2 + b_2 x_i y_i + b_3 y_i^2 + b_4 x_i + b_5 y_i + b_6 \qquad (2)$$
$$i = 1, ..., n$$

where n is the number of transformed points and a_i, b_i are the transformation parameters.

V. THE QUADRATIC DEFORMATION MODEL CONFORMITY WITH THE MPEG-4 FACIAL ANIMATION STANDARD

The derived FDT parameters are captured from the non-linear nature of the muscle deformations using quadratic deformation models. On the other hand, the MPEG-4 Standard was solely developed and standardised based on the FACS AUs. To be able to use the FDT parameters as a mechanism to control and synthesize facial expressions on the top of the MPEG-4 facial animation standard we need to check for the conformity of the quadratic deformation models with the FACS action units (AU) description of muscle movements. In order to do this check the following process is carried out:

1) Generate a set of transformation constraint rules on the MPEG-4 FAPs that correspond to the AUs involved in the main universal expressions E (smile, sad, surprise, anger, fear, and disgust). Fig. 3 illustrates the constraint rules[1] for the transformations of the FAP points involved in each AU.
2) Map the defined muscle based facial regions, by Obaid et al. [3], to the corresponding FAP points as shown in Fig. 4.
3) Apply the quadratic transformation of expression E (FDT_E) to the set of FAP points[2] .
4) Conformity is achieved if the transformed FAPs meets the constraint rules for each AU involved in E.

Using the process described above we have confirmed that the quadratic deformation models do conform to the AUs involved in each of the main universal facial expressions. This

---

[1]Constraint rules are based on the displacement of FAP_i and its transformation FAP_i'.

[2]The FAP points are normalised as described in [3].

Fig. 3. Constraint rules for FAPs based on the AUs involved in the main universal expressions (AU images are adopted from [12][5], while the MPEG-4 FAP images are generated using the FAPs' illustration in [6]).

conformity allows us to utilize the MPEG-4 facial animation parameters with the quadratic deformation model representations (FDT) of facial expressions as illustrated in Fig. 1.

The following section describes how the FAP values are computed from FDTs for different facial expressions.

VI. FDT TO FAP MAPPING

In our approach, we map the defined muscle based facial regions into the corresponding FAP points[2] described by the MPEG-4 facial animation standard as shown in Fig. 4. This allows us to generate different deformations and exaggerations of the FAP points to form the desired facial expression appearance.

For instance, if the FDT parameters of Expression E are applied to the mapping shown in Fig. 4, the position of the FAP points will transform to the desired expression E. However, to be able to deform a 3D facial model compliant with MPEG-4 facial animation standard, the displacement of each point has to be converted into a FAP value that matches the 3D facial
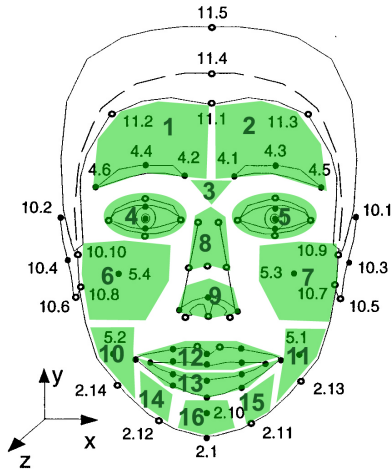
Fig. 4. Facial muscle regions mapped to the corresponding FAP points.

Fig. 5. Mapping between (a) transformed facial points P_i using FDT and (b) its corresponding FAP_i point.

model semantics. The following steps describe the process of converting the displacement of a facial point P_i to a FAP value defined by the semantic rules of a 3D facial model.

- Deform a set of FAP facial points (which we represent as P_i) using the quadratic models as shown in Fig. 4.
- Define the minimum and maximum displacement of the transformed facial points (P_i) in the xy-plane.
- Extract the corresponding minimum and maximum values of FAP_i from the 3D facial model semantic rules.
- Use linear mapping between the xy-plane values and the FAP-plane values to compute the FAP values. This process is described in Fig. 5.
- The derived FAP values can then be used to synthesize facial expressions on the 3D facial model.

The pseudo-code below illustrates how the process shown in Fig. 5 is computed mathematically using linear mapping with respect to the y-axis.

```
Integer getFAPValueY(y : Integer);

vars
 range, deformed_range : Integer;
 fap_range, fap_y_i : Integer;
 ratio : Decimal;
begin
 range = Ymax - Ymin;
 deformed_range = Pi - Ymin;
 ratio = deformed_range/range;
 fap_range = fap_y_max - fap_y_min;
 fap_y_i = (ratio*fap_range) + fap_y_min;

 return fap_y_i;
end;
```

Fig. 8 shows an example of mapping the FDT parameters of several facial expressions to Greta's facial model. The
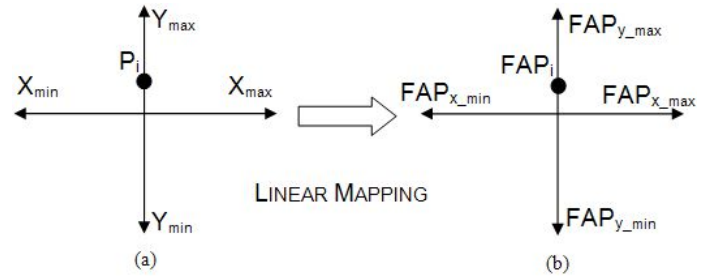
ability to map the FDT parameters of main facial expressions allows us to control the expressivity of the facial features and also generate intermediate facial expressions. The following sections describe in more detail how the expressivity of the facial features is controlled.

VII. EXPRESSIVITY OF THE FACIAL APPEARANCE

The way the FDTs are defined allows us to control the expressiveness of the 3D facial model appearance and its level of exaggeration. This is done by interpolating the FDTs parameter values between the neutral facial expression and the desired facial expression. Extrapolating the parameters beyond the desired expression will generate an exaggeration of the appearance of the facial feature parts. In this context, Equations 1 and 2 can be re-written as follows:

$$x'_{i(E)} = \sum_{u=0}^{2}\sum_{v=0}^{2} A_{uv(E)} x_i^u y_i^v \tag{3}$$

$$y'_{i(E)} = \sum_{u=0}^{2}\sum_{v=0}^{2} B_{uv(E)} x_i^u y_i^v \tag{4}$$

where suffix (E) denotes a transformation specific to an expression E, and $A_{00} = a_6$, $A_{01} = a_5$, $A_{10} = a_4$, $A_{02} = a_3$, $A_{11} = a_2$, $A_{20} = a_1$, with similar mapping for coefficients b_i.

A parametric linear interpolation between a neutral expression N, and a given expression E is then given by

$$x'_i = (1-t)x'_{i(N)} + tx'_{i(E)}$$
$$= \sum_{u=0}^{2}\sum_{v=0}^{2}((1-t)A_{uv(N)} + tA_{uv(E)})x_i^u y_i^v \tag{5}$$

$$y'_i = (1-t)y'_{i(N)} + ty'_{i(E)}$$
$$= \sum_{u=0}^{2}\sum_{v=0}^{2}((1-t)B_{uv(N)} + tB_{uv(E)})x_i^u y_i^v \tag{6}$$
$$0 \leq t \leq 1$$

The parameter t defines the degree of the facial expression E and it is used for producing different facial expressions on the 3D facial model. The value of t could be increased above 1 to extrapolate the coefficients beyond the normal values for that expression. Our approach allows the user to interpolate the parameter t by varying the value, interactively using a slider,

Fig. 6. A facial animation sequence of the surprise facial expression.

between $0 \leq t \leq 1$ or extrapolate the value $t > 1$. A continuous variation of the parameter t produces an animation sequence for the given facial expression. An example of the animation sequence for the facial expression surprise is shown in Fig. 6.

A. FDT for Intermediate Expressions

The ability of using Equations 5 and 6 to interpolate between the facial deformation parameters for an expression E, allows us to generate different intensity levels of the facial expression E. These expressions can be interpreted as intermediate expressions of E or expressions that belong to the same category of E.

From psychology, Whissel [13] studied how different activation levels of an expression can form other facial expressions or emotions. In her studies she defined different categories of a facial expression E (from the six universal expressions) on a two dimensional axes (Activation and Evaluation) as shown in Fig. 7. Activation refers to the level of arousal with respect to E, while Evaluation refers to the level of agreement. Table I shows an example of some of the activation and evaluation values represented by Whissel's study. The evaluation levels of a facial expression can express the internal feelings of an individual, which in turn can be difficult to define using facial muscles. However, the activation level can be directly associated with facial muscles and their movements.

TABLE I
EXAMPLE OF EMOTION ACTIVATION AND EVALUATION VALUES FROM
WHISSEL'S STUDY [13]

Emotion	Activation	Evaluation
Angry	4.2	2.7
Hostile	4	1.7
Furious	5.6	3.7

Using the activation levels (defined by Whissel) we can generate intermediate facial expressions of E by mapping their activation level to the FDT parameters. This can be done by computing the parameter t from Equations 5 and 6 as follows

$$t = (a_I/a_E) \qquad (7)$$

where the a_I and a_E are the activation levels for the intermediate facial expression I and the main facial expression E respectively.
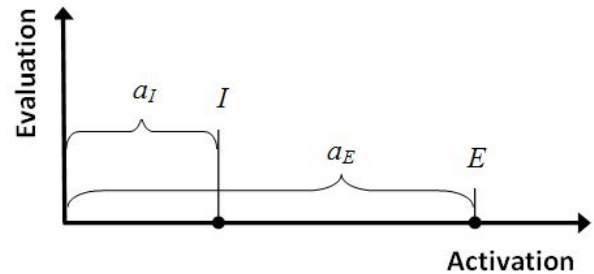

Fig. 7. Whissel's facial expression two dimensional space (Activation and Evaluation).

VIII. RESULTS USING THE GRETA ECA FACIAL MODEL

In this section we illustrate the results of utilizing the Greta ECA engine with our proposed approach. Fig. 8 shows an example of Greta's face expressing all of the main universal facial expressions (smile, fear, disgust, surprise, sad, and anger). Fig. 9 gives an example of Greta showing different intermediate expressions generated using the activation parameters defined by Whissel's study [13] as shown in Table I. Fig. 6 demonstrates an animation of the surprise facial expression achieved by interpolating the FDT parameters (as described in section VII) between the neutral and surprise expressions.

CONCLUSIONS AND FUTURE WORK

We proposed an approach to synthesize and control 3D facial models, compliant with the MPEG-4 facial animation, using quadratic deformation model representations of facial expressions. We conform the quadratic deformation model with the FACS AU for the purpose of utilizing the MPEG-4 facial animation standard with the quadratic representations.

Facial expressions are synthesized by computing the FAP values based on the semantics of the 3D facial model. The FAP values for facial expression E are computed by performing linear mapping between a deformed set of FAP points (using FDT) and the 3D facial model semantics. We use the activation level of Whissel's study [13] along with our quadratic deformation model interpolation technique to generate not only the main universal expressions but also intermediate expressions. The achieved results are encouraging and can lead to future work in the field of expressive 3D facial models.

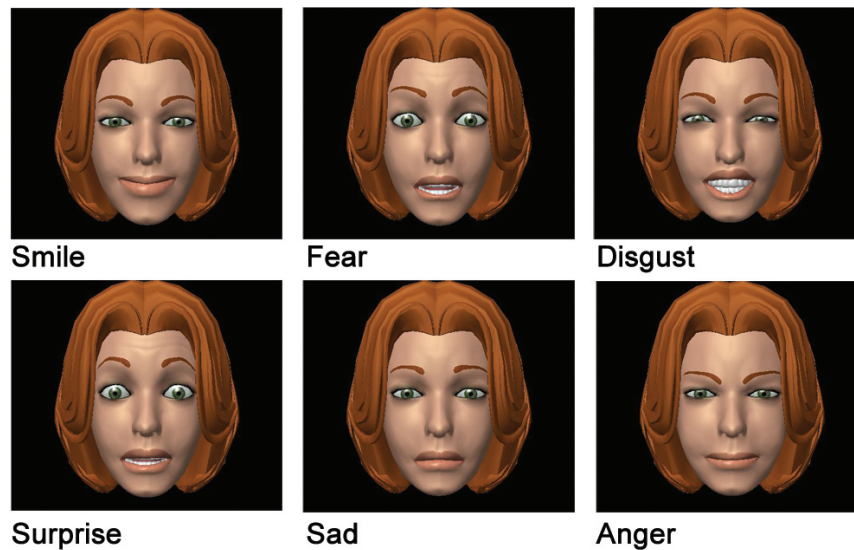Our future directions for this research are focused on following:

Fig. 8. Greta expressing the six main universal facial expressions described by Paul Ekman [14].

Fig. 9. Intermediate expressions of the Anger Facial Expression.

- Further enriching the type of expressions that can be generated using the quadratic deformation models.
- Use of the emotion wheel described by Plutchik [15] to generate facial expressions belonging to two different emotions.
- Blending two facial expressions together using the quadratic deformation models.
- Conducting subjective evaluations of the facial expression synthesized on the 3D facial models using our approach.

ACKNOWLEDGEMENTS

REFERENCES

[1] V. Vinayagamoorthy, M. Gillies, A. Steed, E. Tanguy, X. Pan, C. Loscos, and M. Slater, "Building expression into virtual characters." in *In: Eurographics Conference State of the Art Reports*, 2006.

[2] M. Argyle, *Bodily communication*. Methuen, 1988.

[3] M. Obaid, R. Mukundan, M. Billinghurst, and M. Sagar, "Facial expression representation using a quadratic deformation model," in *CGIV 2009: International Conference on Computer Graphics, Imaging and Visualization*. Tianjin, China: IEEE Computer Society, 2009.

[4] S. Pasquariello and C. Pelachaud, "Greta: A simple facial animation engine," in *In Proc. of the 6th Online World Conference on Soft Computing in Industrial Applications*, 2001.

[5] P. Ekman and W. Friesen, "Facial Action Coding System (FACS): Manual," *Palo Alto: Consulting Psychologists Press*, 1978.

[6] I. S. Pandzic and R. Forchheimer, Eds., *MPEG-4 Facial Animation: The Standard, Implementation and Applications*. New York, NY, USA: John Wiley & Sons, Inc., 2003.

[7] F. I. Parke and K. Waters, *Computer Facial Animation*. AK Peters Ltd, 2008.

[8] M. Radovan and L. Pretorius, "Facial animation in a nutshell: past, present and future," in *SAICSIT '06: Proceedings of the 2006 annual research conference of the South African institute of computer scientists and information technologists on IT research in developing countries*. Republic of South Africa: South African Institute for Computer Scientists and Information Technologists, 2006, pp. 71–79.

[9] A. Paradiso, "An algebra for combing mpeg-4 facial animations," in *In proceedings of the Workshop on Life-like characters: Tools, Affective Functions and Applications*, August 2002.

[10] A. Raouzaiou, N. Tsapatsoulis, K. Karpouzis, and S. Kollias, "Parameterized facial expression synthesis based on mpeg-4," *EURASIP J. Appl. Signal Process.*, vol. 2002, no. 1, pp. 1021–1038, 2002.

[11] R. Gonzalez and R. Woods, *Digital Image Processing*, 3rd ed. Prentice-Hall, 2007.

[12] Carnegie Mellon University, Rebotic Institute, Automated Face Analysis Group. "FACS - Facial Action Coding System". Retrieved September 26th, 2009, from http://www.cs.cmu.edu/~face/index2.htm.

[13] C. M. Whissel, "The dictionary of affect in language," in *In Emotion: Theory, Research and Experience*, R. Plutchik and H. Kellerman, Eds., vol. 4. Academic Press, 1989, pp. 113–131.

[14] P. Ekman, *Unmasking the face*. Prentice Hall., 1975.

[15] R. Plutchik, *Emotion: A Psychoevolutionary Synthesis*. Harper and Row, 1980.