Face Tracking and Realistic Animations for Telecommunicant Clones

Stéphane Valente and Jean-Luc Dugelay
Eurécom Institute

Using realistic face models and photometric modeling techniques, we present a visual feedback loop that tracks a face—without any marker or controlled lighting—throughout a video sequence and precisely recovers the face position and orientation. We also propose animation techniques to embed realistic expressions in our 3D clones. Such face models permit automatic construction of appearance models.

The ability to analyze a human's facial expressions in a video sequence and reproduce them on a synthetic head model is important for many multimedia applications such as model-based coding, virtual actors, human-machine communication, interactive environments, video telephony, and virtual teleconferencing. The literature features three general analysis and animation techniques that can perform this task, depending on how the synthesis parameters relate to the analysis parameters:

1. *Feature-based techniques and animation rules.*

   These methods build on parametric face models, animated by a few parameters to directly control the properties of facial features like the mouth aperture and curvature or the rotation of the eyeballs. The analysis strategy consists of measuring some quantities on the user's face such as the size of the mouth area using blobs, snakes, or dot tracking. Animation rules then translate the measurements in terms of animation parameters.[1,2] For example, some algorithms already meet real-time analysis frame rates, while allowing performers some degree of freedom in their head position and orientation.[3,4]

   The first problem with this approach is that the measurements extracted from the real images are rather sparse and don't allow an accurate animation of the synthetic face model to reproduce subtle expressions. A second problem is that whenever a new animation parameter is integrated into the framework, a new analysis method must be implemented with the adequate animation rule on a case-by-case basis. The last problem is that controlled lighting, makeup, or markers are needed to avoid any ambiguity on the measured parameters, which could dramatically affect the face animation system.

2. *Analysis-by-synthesis techniques and wireframe adaptation.*

   Instead of using sparse measurements like dots, edges, or color areas, the dense motion information—computed on the user's face—is interpreted in terms of displacements of the face model wireframe via an analysis-synthesis feedback loop. The face model can be either parametric or muscle-based.[5,6]

   The main advantage of these techniques is that they provide a single and automatic framework to integrate all the possible animation parameters as degrees of freedom during the nonrigid motion regularization while handling the task of determining the head pose as global degrees of freedom. They yield precise animations when a realistic face model interprets the velocity information. However, such algorithms don't achieve real-time performance because of the iterative feedback loops that at each time step must linearize the 3D motion of the wireframe in the 2D image plane where the optical flow is extracted.

3. *View-based techniques and key-frame interpolation.*

   Like the former algorithms, view-based techniques rely on dense information, taking into account the distribution of the pixels in the analyzed images, but avoiding the iterative procedures needed to solve the animation parameters. The face animation occurs by interpolating the wireframe between several predefined configurations (keyframes), representing some extreme facial expressions. The difficulty lies in relating the performer's facial expressions to the keyframes and finding the right interpolation coefficients from the real image. Appearance models of the distribution of pixel intensities around the facial features generally achieve this solution. These models character-

1070-986X/00/$10.00 © 2000 IEEE

ize the facial expressions according to a database of example pairs of images and animation weights and are used to train the system.

In Essa et al.[5] template-matching algorithms compute correlation scores with examples found in a small database. Radial basis function (RBF) interpolators (a specific class of neural networks) produce the interpolation coefficients from the correlation scores. Another example of appearance models comes from Ohya, Ebihara, and Kurumisawa,[7] where the discrete cosine transform (DCT) coefficients of the analyzed facial features are transformed into interpolation weights by a linear matrix, built on a training database using genetic algorithms. (Some markers are pasted on the performer's face to make the DCT coefficients more significant.)

Although view-based techniques and key-frame interpolation are quite intuitive and remain the preferred methods for real-time performance, they suffer from several difficulties. First, the appearance models must be carefully designed to take into account the coupling between the head pose (the 3D position and orientation of the user's face) and the facial expressions. (For instance, if performers nod their heads downward, their mouths will be curved in the image plane, and you could falsely interpret it as a smile.) This is mainly why these algorithms generally require performers to stay in a strict frontal view with respect to the acquisition camera.

Second, the quality of the training database limits such a system, since it requires a real user to mimic the facial expressions of the face model in front of a camera according to the example synthesis parameters. (Or, the user must manually find the right interpolation weights on the face model to reproduce the facial expressions found in a set of real images.) Ideally, all the sample images should be in the same referential (the real user must not move between different samples). The training database must include all the degrees of freedom of the face model to sample the variability of the face for complex facial expressions (such as a neutral closed mouth, a neutral open mouth, a smiling closed mouth, a sad open mouth, and so on, requiring the user to perform thousands of facial expressions in the same referential).

Finally, the sample images and the animation parameters must be carefully related, since each pair has to correspond to the real and synthetic facial expressions with the same exact intensity. That is, users need to precisely control their facial expressions according to the face model, which proves more difficult when the face model lacks realism. Needless to say, meeting all these training constraints with a real user is impossible.

Visual analysis-synthesis cooperations

Unfortunately, in the literature only a few face cloning algorithms take advantage of the visual realism of their face model to track and/or analyze facial deformations while minimizing the amount of information processed. In most analysis-synthesis feedback loops, the face model's geometric data is explicitly manipulated, linearized in the image plane, and used to solve iteratively for the rigid and/or nonrigid face displacements.[5,6,8]

As we describe in this article, a better approach to reaching real-time analysis frame rates is to manipulate pixels instead of 3D primitives and design the cooperation between analysis and synthesis modules at the image level. In this case, the burden of the 3D manipulation of the face model and the 2D conversion is translated to dedicated graphics hardware, available on most entry-level workstations and PCs. We call this approach a visual analysis-synthesis feedback loop.

Researchers have recently investigated analysis-synthesis cooperations using realistic face modeling and graphics hardware. Schödl, Haro, and Essa[9] discussed projecting the first image of a real user's face onto a generic face model. Using a steepest-descent algorithm, they mapped the derivates of the error function between the analyzed real image and the synthetic model view to track the face in subsequent frames. La Cascia, Isidoro, and Sclaroff[10] provided another example of efficient face tracking. They described projecting the initial face image onto a cylinder. Tracking then occurs by the registration of the face texture in the cylindrical texture map, which modifies the cylinder's position and orientation parameters in the real image. However, these visual analysis-synthesis cooperations need several iterations before they converge to a local minimum.

Overview of our work

Here we present our face cloning research in the Traivi (Traitement des Images Virtuelles, or processing of virtual images) project,[11] which aims to build a virtual teleconferencing space. Three-dimensional face models represent people (see our author photos for an example) and reproduce their rigid and nonrigid motion at distant sites (hence our use of the name telecommunicant clones). Since we aim to provide a high level

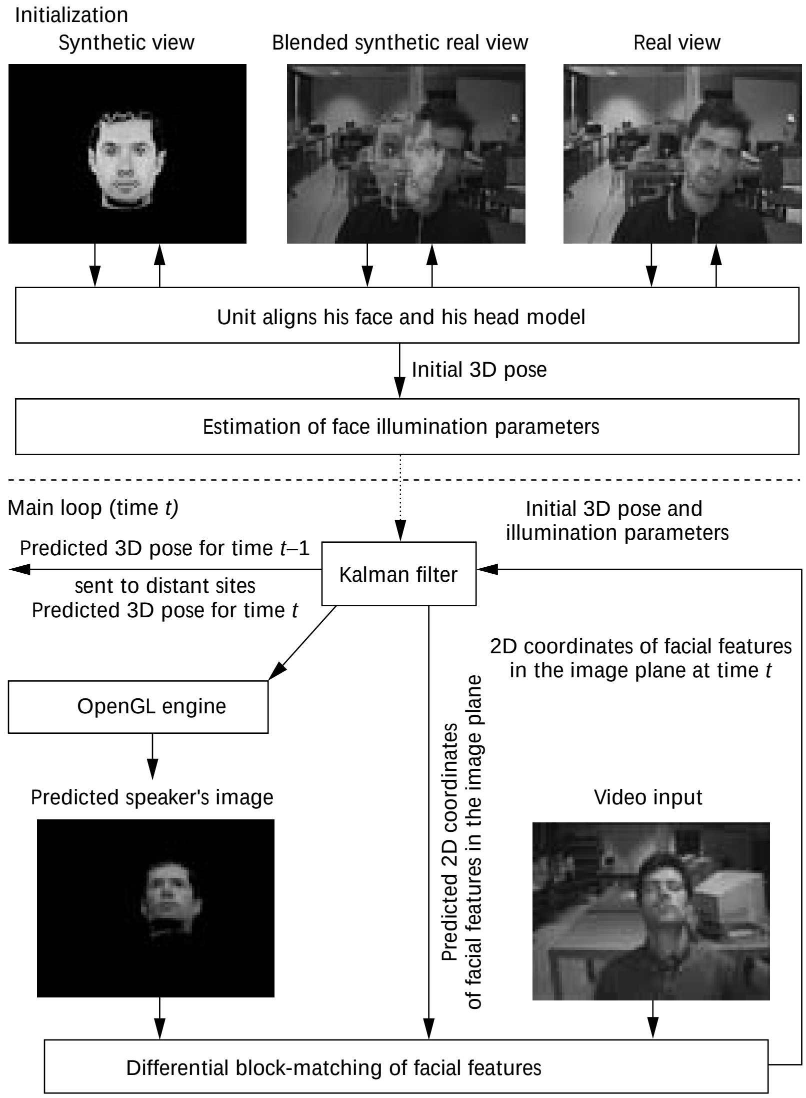


Figure 1. System overview. The initialization and face tracking loop.

of realism and impose as few constraints as possible on the users (like markers, makeup, controlled lighting, or restricted motion in front of a camera), we prefer researching visual analysis-synthesis techniques. This approach leads to efficient cooperation in the image plane and make up for the limited constraints that we target. Currently, our realistic clones are person-dependent face models built from Cyberware range data (see http://www.cyberware.com), but any realistic face model could be integrated in our algorithms.[12]

Head pose determination

We first addressed the problem of tracking the face and determining its 3D rigid motion in our face cloning system by using an analysis-synthesis cooperation. We designed a system that proceeds as follows (Figure 1):

1. Initialization

- Users align their heads with their head model, or alternatively modify the initial pose parameters to align their face model with their heads. Although this step requires some user intervention, it's performed only at the beginning of the session. Automatic initialization procedures could also be applied, using eigenfeatures[13] or frame-fitting.[14]

- When done, the system runs a 3D illumination compensation algorithm to estimate the lighting parameters that will reduce the photometric differences between the synthetic face model and the real head in the user's environment. To compensate for the differences, the system adjusts the intensities of synthetic lights set at arbitrary locations in the 3D world (as described and justified elsewhere[15]).

2. Main loop

- A Kalman filter predicts the head's 3D position and orientation for time *t*.

- The synthetic face model generates an approximation of the way the real face will appear in the video frame at time *t*. This approximation includes geometric distortions, the scale and shaded lighting due to the speaker's pose, and some clues about the location of the background with respect to the face's tracked regions.

- Patterns representing contrasted facial features (like the eyes, eyebrows, mouth corners, and nostrils) are extracted from the synthesized image.

- An extended differential block-matching algorithm matches these patterns with the user's facial features in the real video frame.[15]

- The system passes the 2D coordinates to the Kalman filter, which estimates the current head's 3D position and orientation.

The visual feedback loop's strength is that it implicitly takes into account the changes of scale, geometry, lighting, and background with almost no overload for the feature-matching algorithm.[16] Because the synthesis module performs a 3D illumination compensation scheme, the synthesized patterns will predict the geometric deformations, lighting, and background location of the user's facial features, making the differential block-matching stage more robust. The tracking algorithm doesn't explicitly manipulate 3D primitives, but it does manipulate 2D synthetic image patches. In addition, compared to other systems,[9,10] ours requires no iterative procedures.

This enhanced analysis-synthesis cooperation results in a stable face tracking framework without artificial marks highlighting the facial features, supports very large rotations out of the image plane (see Figure 2), and even copes with low-contrast lighting due to the 3D illumination modeling (as shown in the MPEG demo available at http://www.eurecom.fr/~image/TRAIVI/valente-8points.mpg). We assessed the tracking algorithm's accuracy on a synthetic video sequence, where the parameters to recover are precisely known (see Valente and Dugelay[17] for more details). The accuracy of the recovered parameters is about 0.5 percent of the face size in translation and two degrees in rotation.

Figure 2. Head rotations supported by the face tracking system, without any marker or specific lighting.

Facial expressions

As argued in the introduction, on the one hand, analysis-synthesis cooperations usually require iterative procedures to estimate the nonrigid motion of a real face. On the other hand, view-based analysis techniques estimate facial expressions in a single step, but are limited by the number and quality of training keyframes obtained from a real user. We therefore propose replacing real users during the training stage of a view-based framework with their realistic clones to obtain better training conditions for the system.

Figure 3 shows this framework. Using a person-dependent clone, we optimally sample the visual space of facial expressions, via an animation database (a collection of animation parameters $\vec{\mu}$), to produce a synthetic image database (a collection of image samples ***I***). All degrees of freedom permitted by the synthetic face can precisely, automatically, and systematically be exploited in the training strategy while taking into account the coupling between several facial animation parameters (FAPs) modifying the same areas of the face. Some features, like plain image patches or optical flow, are extracted from the images to represent the facial expressions corresponding to the animation database. Then, we perform a dimensionality

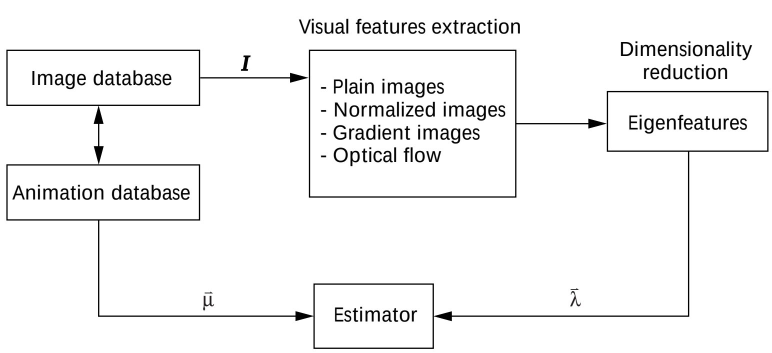


Figure 3. Training framework for the analysis of facial expressions on ($\vec{\mu}$,$\vec{\lambda}$) example pairs.

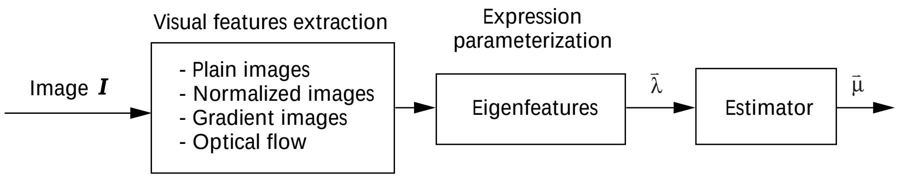


Figure 4. Analysis of an unknown expression.

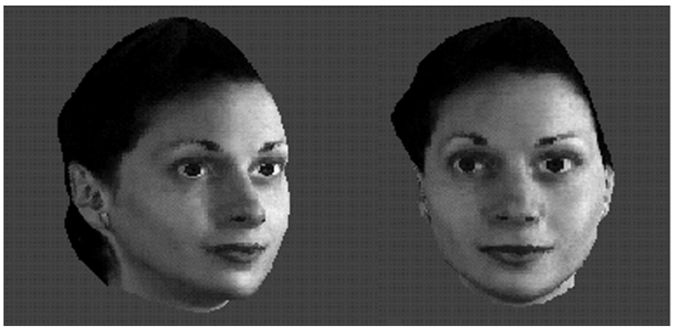


Figure 5. Katia's original face model.

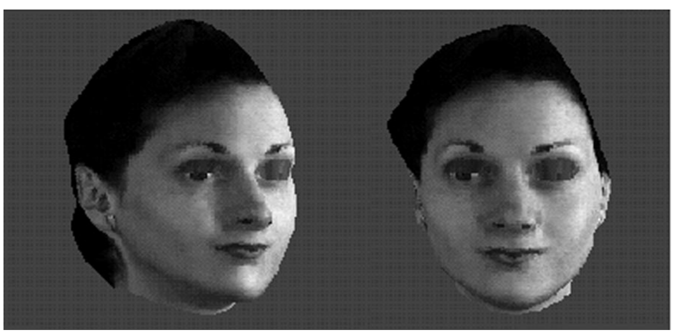


Figure 6. Displacements of mesh vertices. The eye on the right is closed, the left one is half-closed, and the mouth is squeezed.

reduction over those features in order to extract a limited number of vectors optimally spanning the variability space. These vectors (called eigenfeatures) will let us characterize the facial expression of the user's face via a simple correlation mechanism, yielding a compact parameterization $\bar{\lambda}$ vector.[18] At last, a clone-dependent estimator learns the relationship between the animation parameters and the facial expressions, built from the collections of $\bar{\lambda}$ and $\bar{\mu}$ vectors. This synthetic training stage can be viewed as another analysis-synthesis cooperation, taking place in the image plane only.

After the synthetic clone trains the system, the analysis procedure extracts the corresponding features (plain image patches, optical flow, and so on), parameterizes them with their eigenfeatures, and interprets them with the corresponding estimator (see Figure 4).

Synthesis of facial expressions

Most studies of 3D face model construction in the literature tried to adapt a more-or-less generic face model to an individual from photographs or range data. Constructing facial animations then becomes straightforward because they're built directly in the generic head model by defining wireframe deformations. As always, a tradeoff exists between real-time rendition capabilities and realism, and the face model may end up being an oversimplified, unrealistic avatar. Instead of starting from a generic model to make it specific to a given person, we took the opposite approach, starting from person-dependent data (range and texture image) corresponding to a neutral facial expression and processing it to make it suitable for a general analysis-synthesis framework. The main difficulty is that, though highly realistic, our face model comes unanimated—it's made of static vertices, attached to a static texture image via static texture coordinates. Another difficulty is that no separate primitives exist for the eyeballs—the initial face model is just a plain surface. Nevertheless, this section will show how to achieve facial expressions by applying simple deformations, not only on the wireframe vertices, but also at the three different levels (vertices, texture coordinates, and texture image), implementing well-known or original animation techniques.

To emphasize that our animation methods are valid in 3D, the next figures will display two points of view of the same model with different facial expressions. That is, they can be displayed under any point of view as required by a virtual teleconferencing system. For future comparisons, Figure 5 gives the initial face model in a neutral facial expression.

Mesh animations. Key-frame animation (or mesh morphing) consists of interpolating the positions of the mesh vertices between extreme facial expressions. It particularly suits real-time and performance animation because it only involves linear combinations between predefined vertex positions. Key-frame animation also smoothly deforms a surface as complex and pliable as the human face. It generally produces less undesirable effects (like bulging, creasing, and tearing) than facial animation created with bones

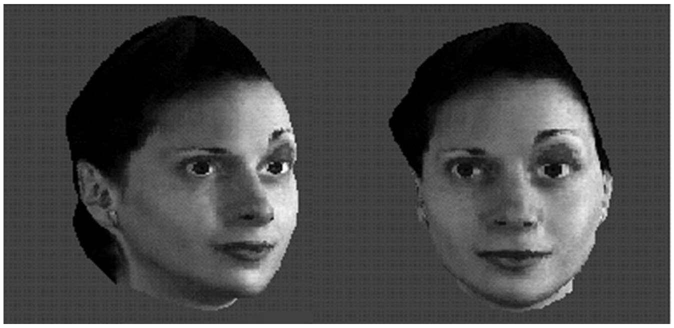


Figure 7. Displacements of texture coordinates. The eyebrow on the left is down, the eyebrow on the right is up.

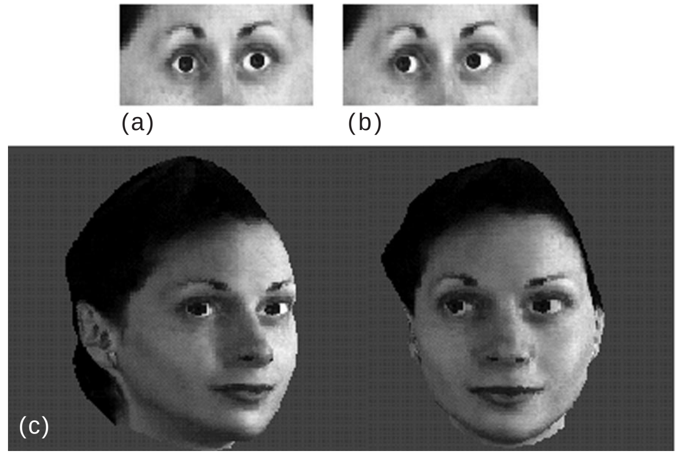


Figure 8. Texture-shifting in the texture plane: (a) Neutral position of the eyes in the cylindrical texture, (b) right shift of the eyes, and (c) result after 3D mapping.

or lattices. We implemented this technique in our face models to animate the eyelids and the mouth (Figure 6).

Texture coordinate displacements. Not all animations require deforming the face model's shape. For instance, lifting an eyebrow corresponds to the underlying muscles sliding up onto the skull. We mimicked this operation by extending the principle of key-frame mesh interpolation to texture coordinates to make the texture image slide over the wireframe.

Figure 7 shows that this technique can implement the motion of the eyebrows correctly. It simulates the extension of the skin just below the right eyebrow by pulling up the eye's makeup while keeping the head shape unaltered. Such an effect would be impossible to achieve by mesh morphing alone.

Texture animations by texture displacements. We alter the cylindrical texture mapped onto the mesh vertices at rendition time to produce further animations. In most face models, the gaze is controlled by the rotation of eyeballs appearing through holes created for the eyes in the wireframe. Instead of adding new primitives for each eye, we created holes in the texture image (via the transparency channel). Two separate textures behind the main one can be displaced to alter the model's gaze direction (Figure 8). Because of the alpha channel, the shape of the eye contours remains unchanged and covers the moving texture portions. We're also using this technique to implement the model's teeth or tongue by overlapping several texture portions on a plane just behind the model's lips. This solution has the advantage of being more realistic than generic primitives accounting for the teeth of an individual person.

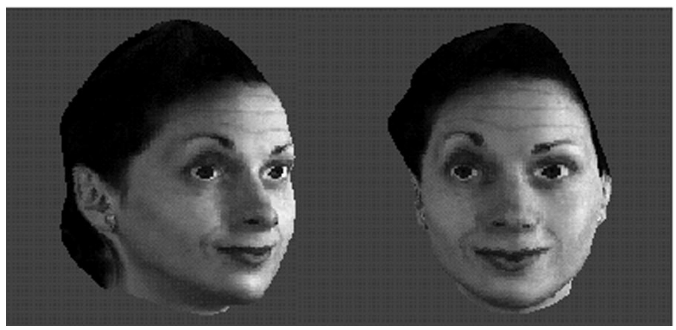


Figure 9. Expression wrinkles and furrows by texture blending.

Texture animations by texture blending. Besides moving some texture portions, it's possible to blend several textures together to produce a new one. For example, you can fade wrinkles into the model texture at a low cost in terms of real-time animation, instead of hard-coding them in heavy spline-based meshes, as seen in Figure 9.

Realistic animations. Each defined model alteration is controlled by a single parameter μ_i, a FAP (conforming to the guidelines of the Moving Pictures Expert Group's MPEG-4 standard).[19] The choice of the animation method depends on the type of FAP implemented, whether the overall shape of the face model must be modified (mesh morphing) or only its skin has to be displaced (texture animations).

Combining n parameters (that is, n independent mesh or texture modifications) in a single vector $\bar{\mu} = (\mu_1,\ldots,\mu_n)^T$, the face model is then capable of complex facial expressions. Although con-

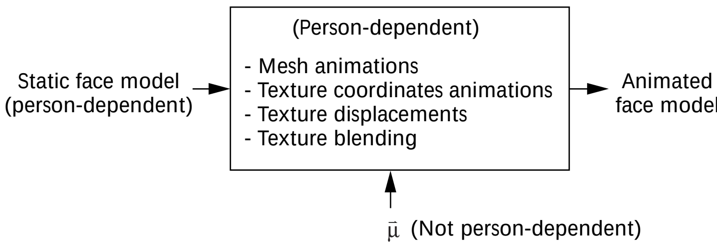


Figure 10. The synthesis part of facial animations. Transforming a static model into an animated one, controlled by the general $\bar{\mu}$ vector.

structing the deformations remains highly person-dependent, facial expressions are controlled by the $\bar{\mu}$ vector, which is completely transparent for the analysis and synthesis frameworks (Figure 10).

If the $\bar{\mu}$ vector relates to the same FAPs across different face models—even though each FAP is implemented in a strict person-dependent manner—it will be possible to analyze the facial expressions of performers using their own model (like in Figure 1) and clone-dependent estimator, and reproduce them onto another 3D model. A video sequence shows a stream of complex facial expressions ($\bar{\mu}$ vectors) on one of our clones at http://www.eurecom.fr/~image/TRAIVI/animation.mpg.

Reproduction of facial expressions: From $\bar{\lambda}$ to $\bar{\mu}$

Once the training databases of visual measurements $\bar{\lambda}$ and animation vectors $\bar{\mu}$ have been generated using a person-dependent clone, we build an estimator to learn the relationship $\bar{\lambda} \to \bar{\mu}$.

Estimators. We investigated two types of estimators:

1. a linear one, modeling the mapping as a linear matrix **L** such as $\bar{\mu} = L\bar{\lambda}$, and

2. an RBF network, synthesizing the mapping in terms of simpler functions centered around the N training examples $\bar{\lambda}$ such as $\bar{\mu} = \sum_{i=0}^{N} c_i \, \lambda^T \bar{\lambda}_i$

   where c_i are the respective weights of the basis functions $\lambda^T \bar{\lambda}_i$. Each input of the network corresponds to the correlation between the analyzed image (modeled by $\bar{\lambda}$) and a training example ($\bar{\lambda}_i$).

The RBF formulation has an interesting property in terms of further complexity reduction. To avoid handling all the training samples (typically a few thousand) in the network, we apply another principal component analysis on the training vectors $\bar{\lambda}$ to extract a limited number of "center" vectors (a few hundred). Elsewhere,[20] we showed that in theory, the performance of the network remains the same for basis functions based on scalar products.

Early experiments. To evaluate the performance of our view-based analysis algorithm and how it can extrapolate the training to new facial expressions, we conducted some preliminary experiments on synthetic data. We provided the system with images of facial expressions corre-

Web Extras

To view the demos mentioned in this article, visit *MultiMedia*'s Web site at http://computer.org/multimedia/mu2000/u1toc.htm and click on the following links:

Visual analysis-synthesis cooperations permit an efficient face tracking algorithm without any markers pasted on the user's face and without any constraints or assumptions on the analyzed view such as the scene lighting, the camera's internal and external parameters, and the face's motion: http://computer.org/multimedia/mu2000/extras/u1034x1.mpg

Simple model deformations (mesh animations, texture coordinate displacements, and texture animations) can be blended together to produce complex, realistic, and person-dependent facial expressions on our clones: http://computer.org/multimedia/mu2000/extras/u1034x2.mpg

Early experiments (without any stabilization post-processings) concerning the analysis of real facial expressions and their reproduction on the corresponding clone: http://computer.org/multimedia/mu2000/extras/u1034x3.mpg

For more information contact Jean-Luc Dugelay at Jean-Luc.Dugelay@eurecom.fr. For future updates of the Traivi project, visit http://www.eurecom.fr/~image.

sponding to a set of $\bar{\mu}$ vectors not included in the training databases. We found that both estimators give similar results. Our framework is presently able to recover the animation parameters with a 10-percent accuracy between the original $\bar{\mu}$ vectors and the estimated ones from $\bar{\lambda}$ measurements.[20]

One predictable issue about our framework is how the training, performed on synthetic images in uniform lighting, can be transposed to real images with unknown lighting. We tested different preprocessings to find lighting-independent features such as normalized or gradient images, or even optical flow fields. In our evaluation,[20] a basic optical flow algorithm[21] provides fairly good visual features to analyze real facial expressions (see Figure 11) from a synthetic training database. The apparent motion field isn't computed between successive frames, but between the initial face image (representing a neutral expression) and the analyzed one. This results in some difficulties in terms of optical flow stability, as you can see in the video sequence at http://www.eurecom.fr/~image/TRAIVI/sv-analyse-reelle.mpg. However, our optical flow implementation could be improved greatly or other image features (like images in a normalized color space) could straightforwardly be integrated in the architecture.

Conclusion

Our development of visual analysis-synthesis cooperations relied on pixels as opposed to 3D vertices manipulations. We avoided iterative procedures and took advantage of hardware accelerations whenever possible. Following this paradigm, it's possible to efficiently track a face in a video sequence and accurately estimate its rigid motion from a single camera, if you use a realistic face model of the person to be tracked.

Our current goals include refining our optical flow algorithm to extract the nonrigid motion characterizing the facial expressions of a real user. We also plan to integrate the possible coupling between the pose of the face and its facial expressions in the cloning system. Our $\bar{\lambda}$ measurements are general enough to incorporate the user's position and orientation parameters as additional vector components. By training the system to "see" the relationship between the global pose and the appearance of facial expressions, the estimator translating the observed $\bar{\lambda}$ vectors into animation parameters $\bar{\mu}$ could automatically take into account the coupling. MM

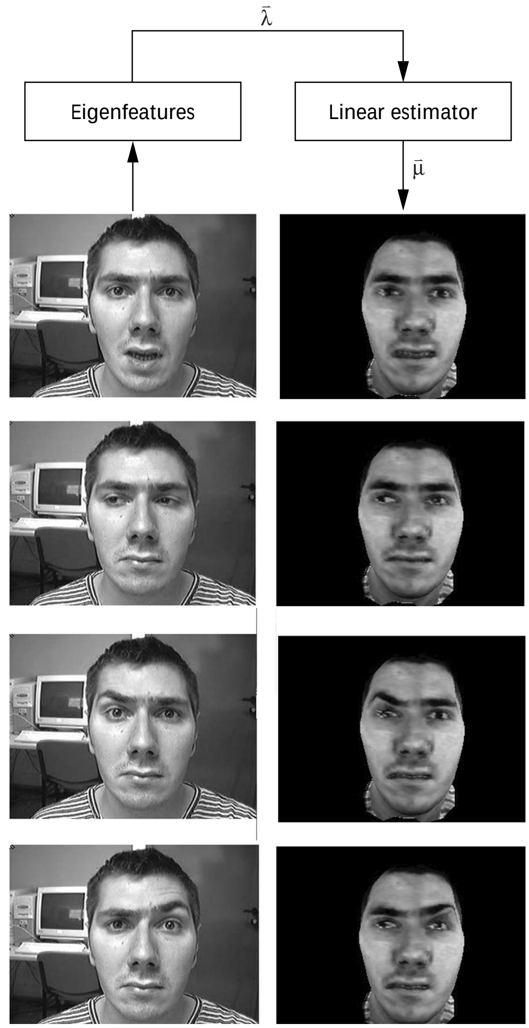


Figure 11. Early analyses of facial expressions. The optical flow field between each image of the left column and the initial (neutral) face image was quantified by some eigenfeatures, giving a $\bar{\lambda}$ vector. A linear estimator mapped the measured $\bar{\lambda}$ to the animation parameters $\bar{\mu}$, which were rendered into the images of the right column.

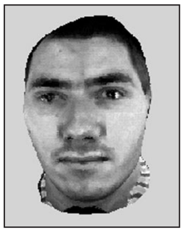
Acknowledgments

Eurécom's research is partially supported by its industrial members—Ascom, Cegetel, France Telecom, Hitachi, IBM France, Motorola, Swisscom, Texas Instruments, and Thomson CSF.

We'd like to thank the University of Erlangen, Germany and the Laboratoire Universitaire d' Applications à la Physique (LUAP, or University Laboratory of Physical Applications at the University of Paris-VII) for the original Cyberware scans and Katia Fintzel (Espri Concept/Eurécom Institute) for being the model in the section "Synthesis of facial expressions."

References

1. D. Terzopoulos, “Modeling Living Systems for Computer Vision,” *Proc. 14th Int’l Joint Conf. Artificial Intelligence*, Morgan Kaufmann, San Francisco, Aug. 1995, pp. 1003-1013.
2. T.S. Huang, S.C. Reddy, and K. Aizawa, “Human Facial Motion Modeling, Analysis, and Synthesis for Video Compression,” *Proc. SPIE Visual Communications and Image Processing 91: Visual Communication,* SPIE Vol. 1605, SPIE Press, Bellingham, Wash., 1991, pp. 234-241.
3. A. Saulnier, M.-L. Viaud, and D. Geldreich. “Real-Time Facial Analysis and Synthesis Chain,” *Proc. Int’l Workshop Automatic Face and Gesture Recognition*, 1995, pp. 86-91.
4. B. Bascle and A. Blake. “Separability of Pose and Expression in Facial Tracking and Animation,” *Proc. Int’l Conf. Computer Vision*, IEEE CS Press, Los Alamitos, Calif., Jan. 1998.
5. I.A. Essa et al., “Modeling, Tracking, and Interactive Animation of Faces and Heads Using Input from Video,” *Proc. Computer Animation 96 Conf.*, Springer-Verlag, Berlin, June 1996.
6. H. Li, P. Roivainen, and R. Forchheimer, “3D Motion Estimation in Model-Based Facial Image Coding,” *IEEE Trans. Pattern Analysis and Machine Intelligence*, Vol. 15, No. 6, June 1993, pp. 545-555.
7. J. Ohya, K. Ebihara, and J. Kurumisawa, “Artistic Anatomy-Based, Real-Time Reproduction of Facial Expressions in 3D Face Models,” *Proc. IEEE Int’l Conf. Multimedia Computing and Systems*, IEEE Press, Piscataway, N.J., June 1999, pp. 684-689.
8. R. Koch, “Dynamic 3D Scene Analysis through Synthesis Feedback Control,” *IEEE Trans. Pattern Analysis and Machine Intelligence*, Vol. 15, No. 6, June 1993, pp. 556-568.
9. A. Schödl, A. Haro, and I.A. Essa, “Head Tracking Using a Textured Polygonal Model,” *Proc. Workshop Perceptual User Interfaces*, Nov. 1998, http://www .research.microsoft.com/PUIWorkshop/Proceedings/Proc-Start.htm.
10. M. La Cascia, J. Isidoro, and S. Sclaroff, “Head Tracking via Robust Registration in Texture Map Images,” *Proc. Int’l Conf. Computer Vision and Pattern Recognition (CVPR 98)*, IEEE CS Press, Los Alamitos, Calif., 1998, pp. 508-514.
11. J.-L. Dugelay, K. Fintzel, and S. Valente, “Synthetic/Natural Hybrid Video Processings for Virtual Teleconferencing Systems,” *Proc. Picture Coding Symp.*, April 1999.
12. M. Proesmans and L. Van Gool, “One-Shot 3D Shape and Texture Acquisition of Facial Data,” *Audio- and Video-Based Biometric Person Authentification*, Crans-Montana, Switzerland, March 1997, pp. 411-418.
13. T.S. Jebara and A. Pentland, “Parametrized Structure from Motion for 3D Adaptive Feedback Tracking of Faces,” *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, IEEE CS Press, Los Alamitos, Calif., Nov. 1996.
14. P.M. Antoszczyszyn, J.M. Hannah, and P.M. Grant, “Accurate Automatic Frame Fitting for Semantic-Based Moving Image Coding Using a Facial Code Book,” *Proc. IEEE Int’l Conf. Image Processing*, IEEE Press, Piscataway, N.J., Sept. 1996.
15. S. Valente, J.-L. Dugelay, and H. Delingette, “An Analysis/Synthesis Cooperation for Head Tracking and Video Face Cloning,” *Proc. Workshop Perception of Human Action* (European Conf. Computer Vision or ECCV 98), Lecture Notes in Computer Science, Springer-Verlag, Berlin, no. 1406 and 1407, June 1998.
16. S. Valente and J.-L. Dugelay, “3D Face Modeling and Encoding for Virtual Teleconferencing,” *Proc. Workshop Very Low Bit-Rate Video (VLBV 98)*, Beckman Inst., Urbana-Champaign, Ill., Oct. 1998.
17. S. Valente and J.-L. Dugelay, *A Visual Analysis/Synthesis Feedback Loop for Unconstrained Face Tracking*, Tech. Report RR-99-051, Mulitmedia Communications Dept., Eurécom Institute, Sophia-Antipolis, France, Nov. 1999.
18. M. Turk and A. Pentland, “Eigenfaces for Recognition,” *J. Cognitive Neuroscience*, Vol. 3, No. 1, 1991.
19. *ISO/IEC 14496-2, Committee Draft ISO/IEC JTC1/SC29/WG11 N1902, Information Technology—Coding of Audio-Visual Objects: Visual*, Int’l Organization for Standardization, Fribourg, Switzerland, October 1997.
20. S. Valente, *Analyse, Synthése et Animation de Clones dans un Contexte de Téléréunion Virtuelle* [*Analysis, Synthesis, and Animation of Clones in a Virtual Telemeeting Context*], PhD thesis, Communication Systems Section, Swiss Federal Inst. of Technology (EPFL), Lausanne, Switzerland, 1999.
21. J.-L. Dugelay and H. Sanson, “Differential Methods for the Identification of 2D and 3D Motion Models in Image Sequences,” *Signal Processing: Image Communication*, Vol. 7, No. 1, 1995, pp. 105-127.



Stéphane Valente received his PhD in communication systems from the Swiss Federal Institute of Technology (EPFL), Lausanne, in 1999, and the French State degree of Ingénieur des Télécommunications (telecommunications engineer) from the National Institute of Telecommunications, Evry, France, in 1994. His research background includes an eight-month internship in speech recognition at the Speech Technology Laboratory (a subsidiary of Panasonic) in Santa Barbara, California, and a 12-month position in the Scientific Data Visualization Department at the Commissariat à l'Energie Atomique (CEA, or Atomic Energy Agency) in Bordeaux, France. His research interests are image and video processing, image synthesis, 3D analysis, and virtual reality. In January 2000, he joined the Ecole des Mines de Paris (Paris School of Mines) as a research engineer.

Jean-Luc Dugelay is a research associate in the Multimedia Communications Department at the Eurécom Institute, Sophia Antipolis, France. He received a PhD in computer science in 1992 from the University of Rennes, France. From 1989 to 1992, he worked at the Centre Commun d'Études de Télédiffusion et Télécommunications (CCETT, or Joint Center for Broadcasting and Telecommunications Studies) in Rennes, for the Centre National d'Etudes des Télécommunications (CNET, or National Center for Telecommunications Studies)/France Telecom R&D Center where he was involved in research on 3D motion and stereoscopic television. His research interests include image processing and coding, watermarking and indexing, video communications, virtual reality, and 3D imaging. He is a member of the IEEE Signal Processing Society Multimedia Signal Processing Technical Committee (IEEE MMSP TC) and is a member of the editorial board of the journal *Multimedia Tools and Applications.*

Readers may contact Dugelay at the Multimedia Communications Dept., Eurécom Institute, BP 193, 06904 Sophia-Antipolis Cedex, France, e-mail Jean-Luc.Dugelay@eurecom.fr, http://www.eurecom.fr/~image.

Contact Valente at the Computer-Aided Design and Robotics Center, Paris School of Mines, 60 Boulevard Saint Michel, 75272 Paris Cedex 06, France, e-mail Stephane.Valente@caor.ensmp.fr.