*Proc. Natl. Acad. Sci. USA*
Vol. 90, pp. 9766–9769, November 1993
Colloquium Paper

***This paper was presented at a colloquium entitled "Images of Science: Science of Images," organized by Albert V. Crewe, held January 13 and 14, 1992, at the National Academy of Sciences, Washington, DC.***

# Image processing: Some challenging problems

T. S. HUANG AND K. AIZAWA

Coordinated Science Laboratory and Beckman Institute, University of Illinois at Urbana-Champaign, 405 North Mathews Avenue, Urbana, IL 61801

**ABSTRACT Image processing can be broadly defined as the manipulation of signals which are inherently multidimensional. The most common such signals are photographs and video sequences. The goals of processing or manipulation can be (*i*) compression for storage or transmission; (*ii*) enhancement or restoration; (*iii*) analysis, recognition, and understanding; or (*iv*) visualization for human observers. The use of image processing techniques has become almost ubiquitous; they find applications in such diverse areas as astronomy, archaeology, medicine, video communication, and electronic games. Nonetheless, many important problems in image processing remain unsolved. It is the goal of this paper to discuss some of these challenging problems. In *Section I*, we mention a number of outstanding problems. Then, in the remainder of this paper, we concentrate on one of them: very-low-bit-rate video compression. This is chosen because it involves almost all aspects of image processing.**

## I. Some Challenging Problems

**Compression.** A modern trend in image storage and transmission is to use digital techniques. Digitizing a television signal results in ≈100 megabits per second. But channel bandwidth is expensive. So for applications such as teleconferencing, one wants to use a channel of 64 kilobits per second. For other applications such as videophone and mobile videophone, even lower channel bandwidths (e.g., 9 kilobits per second) are desirable. How can one compress the bit rate from 100 megabits per second to 1 kilobit per second without severe loss of image quality? One approach to solving this problem will be described in *Sections II–VI*.

**Enhancement.** In enhancement, one aims to process images to improve their quality. An image may be of poor quality because its contrast is low, or it is noisy, or it is blurred, etc. Many algorithms have been devised to remove these degradations. The difficult problem is how to remove degradations without hurting the signal. For example, noise-reduction algorithms typically involve local averaging or smoothing which, unfortunately, will blur the edges in the image. Adaptive methods have been investigated—e.g., smoothing less near the edges. However, they are generally effective only if the degradation is slight. A challenging problem is then how to enhance severely degraded images.

**Recognition.** Typically, a recognition system needs to classify an unknown input pattern into one of a set of prespecified classes. The task is fairly easy if the number of classes is small and if all members in the same class are almost exactly the same. However, the problem can become very difficult if the number of classes is very large or if members in the same class can look very different. Thus, a most challenging problem is how to recognize generic objects. For example, how can one construct a system to recognize "chairs"?

**Visualization.** Commonly, visualization is considered as a part of computer graphics. The main task is to generate images or image sequences based on three-dimensional object and scene models. A challenging problem is how to model dynamic scenes containing nonrigid objects (such as clothing, hair, trees, waves, clouds, etc.). The models have to be realistic, and yet the computation cost has to be reasonable.

## II. Model-Based Video Compression

In the remainder of this paper, we shall discuss in some detail an approach to achieving very-low-bit-rate video transmission or storage. We present some preliminary results of our work on model-based compression of video sequences of a person's face, in the context of teleconferencing and videophone applications. The emphasis is on the difficult and challenging problem of analysis. Algorithms are presented for extracting and tracking key feature points on the face and for estimating the global and local motion of the head/face.

Historically, progress in image coding techniques has been through incorporating results from other fields such as information theory. Most of the existing coding methods such as predictive coding, transform coding, and vector quantization belong to information-theory-based methods, in which image signals are considered as random signals and compressed by exploiting their stochastic properties.

Apart from these information-theoretic coding methods, research on new approaches to image coding which are related to both image analysis and computer graphics has recently intensified. This type of coding method takes into account the three-dimensional nature of the scene. Contrary to conventional coding methods which efficiently represent waveforms of image signals, these new approaches represent image signals by using structural image models. An essential difference between conventional coding methods and these new approaches is the image model they assume. A major advantage of this new coding method is that it describes image content in a structural way. Its application areas are naturally different from those of waveform coding.

Our model-based coding system consists of three main components: a three-dimensional facial model, an encoder, and a decoder. The encoder separates the object from the object's background, estimates the motion of the person's face, analyzes the facial expressions, and then transmits the necessary analysis parameters. The encoder will adapt new depth information and initially unseen portions of the object into the model by updating and correcting it if required. The decoder synthesizes and generates the output images by using a three-dimensional facial model and the received analysis parameters.

**Modeling a Person's Face.** Modeling an object is the most important part in model-based coding because both analysis

The publication costs of this article were defrayed in part by page charge payment. This article must therefore be hereby marked "*advertisement*" in accordance with 18 U.S.C. §1734 solely to indicate this fact.

and synthesis methods strongly depend on the model they use. For communication purposes, a person's face must be modeled in sufficient detail. Previous studies which have attempted to image a human face with graphics have produced results which lack details and reality because the imaging methods that relied on computer graphics techniques used only wire frame models and shading techniques to reconstruct a human face.

In order to develop an accurate model, the original facial image and a texture mapping technique were utilized. That is, a three-dimensional wire frame general face model which approximately represents a human face and which is composed of several hundred triangles is deformed to fit its feature point positions and outmost contour lines to those of a frontal face image of the object person. Then, this original face image is mapped on the adjusted wire frame model. Once a three-dimensional facial model has been obtained it can be easily rotated in any direction. The rotated images still appear natural, although the depth information on the created three-dimensional facial model is only a rough estimate.

**Synthesis of Facial Movements.** Texture mapping of original facial images onto a three-dimensional wire frame model gives rise to natural-looking images. In addition, to synthesize naturally animating images, synthesis of facial movements (expressions) plays a very important role in the model-based coding system. In our system, a facial structure deformation method is used. This method controls the locations of the vertices of the wire frame model according to "deformation rules" that are based on psychological and anatomical knowledge. In our contributions the "Facial Action Coding System" (1) has been adopted, in which facial actions are decomposed into combinations of "action units." Because there are many different ways of deforming a three-dimensional model, other contributions have used different deformation methods (2, 3). In order to improve the reconstruction quality, texture updating is required. Some approaches were proposed which additionally encode differential signal between input images and synthesized images by using DCT (4), VQ (5), and contour coding (3).

**Analysis of Facial Images.** Analysis problems are more difficult than synthesis problems. There seems to be no system, at present, which can work automatically both in modeling objects and in extracting model parameters even for restricted head-and-shoulder images. Some preliminary results of our work in analysis are presented in the following sections.

## III. Facial Motion Analysis

Several algorithms have been proposed to do facial analysis, the most significant ones being the work of Aizawa *et al.* (6, 7), Forcheimer *et al.* (8, 9), Kaneko *et al.* (2, 10, 11), and Keppei and Leidtke (12), but none has used effectively the available three-dimensional information of the face for purposes of the analysis. Neither have they used the psychological research results on the facial expressions. Past research has involved looking only at the two-dimensional images and using a three-dimensional model solely for the purpose of synthesis. Research has also been done in tracking features across a sequence of images, the most significant being the work done on "snakes" by Kass *et al.* (13). However, the second-order contours defined by Terzopoulos *et al.* are capable of tracking features only if the motion is very small, and fails completely for relatively larger motion. Furthermore, none of the research has been able to track a feature and also label the feature as being hidden from the camera and not just label it as being undetectable in the image. We have tried to make use of the available information to track and label features correctly. We have also incorporated the psychological research results to predict and verify the motion detected in the image sequences. Global motion is defined as the motion of the head with absolutely no change in any of the expressions on the person's face. Local motion is only the change in expressions. All local motion can be expressed by vectors called action vectors. Local motion can be completely defined by detecting the location of 26 points called control points. The features of the face are obvious physical features such as the edges of the lips, eyes, and eyebrows. Control points lie on these features. This database of action vectors is used to predict and verify detected features. The feature and hence the control points are tracked across the image sequence by using the information about their possible direction and magnitude of motion as defined by the action vectors. "Search regions" are established around each feature based on this information. Search regions are cuboidal regions in three dimensions and are the orthographic projections of these regions onto the image plane. By transforming these search regions by suitable transformation depending on approximated and then the computed motion, it is possible to clearly define the region in which a feature can be detected, in any image of the sequence. Furthermore, if the face is so transformed that the feature is not visible to the camera, then it will be labeled as being hidden to the camera and hence undetectable. The sections on methodology explain the process of mapping the image onto the three-dimensional model, locating the features, and tracking the features across a sequence of images. The mathematical tools used in obtaining the results are also discussed in this paper. A summary of results and the comparison of results with research done in the past form the sections on results and discussion.

## IV. Methodology

**Mapping.** Candide (14) is a three-dimensional wire frame model of the face. It is defined by a set of vertices $v(x, y, z)$ and a set of planar triangular patches with three of the elements of the vertex set as the vertices. The depth of the head is considerably smaller than the distance between the camera and the face, and so only orthographic projections are assumed. The two-dimensional projection of this model is $v'(x, y)$. The first image in the sequence of images is a direct frontal view of the person. The spatial gradient $S(x, y)$ of this image and the temporal gradient $T(x, y)$ are obtained. From the spatial and temporal gradients, the size of the face in the image sequence is estimated. A threshold for the temporal gradient ($T_{\text{threshold}}$) is specified. Then we define

$$\begin{aligned} F(x, y) &= S(x, y);\ |T(x, y)| \geq T_{\text{threshold}} \\ &= 0;\ |T(x, y)| < T_{\text{threshold}}. \end{aligned}$$

The length and width of the face are estimated from $F(x, y)$. The candide is scaled appropriately along the $x$ and $y$ axes. The scaling factor of the $z$ coordinates of $v(x, y, z)$ is, empirically, approximately the mean of the $x$ and $y$ scaling factors. The origin of the coordinate frame lies in the geometric center of $F(x, y)$.

The mapping of the face onto the model requires a knowledge of the location of the vertices on the model that correspond to the feature points in the image. An interactive procedure specifies this correspondence which results in the translation on some of the vertices of $v'$. This does not, however, affect the $z$ coordinate of $v$. The texture of the face is now mapped orthographically onto the model, thereby generating texture values for the vertices $v$. The mapping is complete with the set $v(x', y', z, \text{texture})$. The texture at each vertex is fixed and does not change throughout the tracking. The vertex texture information is used in the synthesis of the detected motion.

**Features and Control Points.** The features that are tracked are natural facial features—i.e., the eyes, eyebrows, the nose, and the lips. The control points, the tracking of which determines the global and local motion, are points which lie on the edges of these natural facial features. Hence, it is sufficient that the edges of these natural facial features be the features that are tracked over the sequence of images (e.g., the corners of the eyes and the lips, which lie on the edges of the eyes and the lips correspondingly). The corners are the control points and the edges are the features.

Each of the features is represented by a spline, whose internal energy serves to impose a smoothness constraint and pushes the spline toward image features such as lines, edges, and contours. The internal spline energy (13) is represented as

$$S_{\text{int}} = [\alpha(s)|v_s(s)|^2 + \beta(s)|v_{ss}(s)|^2]/2,$$

where $\alpha(s)$ is a first-order term, $\beta(s)$ is a second-order term constituting the internal energy, $v(s) = [x(s), y(s)]$ is the parametric representation of the spline, $v_s = dv/ds$, and $v_{ss} = d^2v/ds^2$. Setting $\beta(s)$ to zero at a point allows the spline to develop a corner. In defining a spline that corresponds to a feature in the image, a discrete formulation is adopted. The above expression is discretized by approximating the derivatives with finite differences and converting to a vector notation with $v_i = (x_i, y_i) = [x(ih), y(ih)]$. By giving different weights to the $\alpha$ and $\beta$ factors, the shape of the spline is made to correspond with the edge of the feature. Hence, setting $\beta_j$ to zero will imply that $v_j$ will always correspond to a corner. The discrete internal spline energy (13) of this feature is

$$S_{\text{int}} \doteq \alpha_i|v_i - v_{i-1}|^2/2h^2 + \beta_i|v_{i-1} - 2v_i + v_{i+1}|^2/2h^4.$$

Also $v_0 = v_n$, since the contours defined are closed contours.

**Tracking.** The methodology we have developed tracks features that may or may not be visible to the camera in the entire image sequence. The splines defined in the previous section are used for tracking the features when they are visible, in whole, and "search regions" are the tools for tracking the features that are wholly or partly hidden from the camera. The splines, in addition to being constrained by their internal energy, are constrained by the external forces, the image $I(x, y)$. We need to constrain the spline to be driven toward the edges in the image. Hence, if we set the external spline energy $S_{\text{edge}} = S_{\text{ext}} = -|\text{laplacian of } I(x, y)|^2$, then the spline is attracted to contours with a large image gradient. The partial derivatives of $S_{\text{edge}}$ are approximated by finite differences. The spline is made to track the feature by minimizing the spline energy $S_{\text{spline}} = \Sigma^n[S_{\text{edge}}(i) + S_{\text{int}}(i)]$. This minimization corresponds to solving two independent Euler equations (13). The result of this minimization is that the spline corresponds to the feature being tracked and that the control points, corresponding to known $\alpha$ and $\beta_i$, are also detected.

A database of vectors, called action vectors (1), each corresponding to the maximum possible local motion of one of the 26 control points, is created. It is this database that forms a part of the motion prediction procedure.

A cuboidal region is defined about the vertices in the model that correspond to a feature in the image. It is oriented in the direction of the normal to the planar patch of the feature. The length of the edge of this cube is such that it would include the control points in this region, which correspond to a 100% displacement along the action vectors. This ensures that the feature will always lie within this region as the region is large enough to account for both the local and the global motion between consecutive frames of the sequence. During tracking, the rectangular region containing the orthographic projection of this cuboidal region is the region in which the search for the entire feature is done. If the feature is not detected, it is labeled as hidden. In reality this feature may only be partially hidden from the camera, but then the spline will not be able to detect this. The action vectors specify the direction in which the spline moves initially when trying to minimize the spline energy $S_{\text{spline}}$. Each set of eight vertices corresponding to the vertices of the cube is appended to $v(x', y', z, \text{texture})$, with the vertex texture information for these appended vertices set to zero. Any transformation performed on the vertices of the three-dimensional model will also transform the cuboidal search regions. Hence, we always know the region in which to look for the features. Even if the feature were completely hidden from the camera, the vertices will define the region in which to look for the feature and, if it is not found, it will label the feature as hidden. As a result, after several transformational operations on the set of vertices $v(x', y', z, \text{texture})$, if a hidden feature is now completely visible, then the region in which it can be found is immediately known.

**Motion Estimation.** For the purpose of motion estimation, only 5 of the 26 control points need be found. Some of the control points have "zero" action vectors; i.e., they are fixed points on the face. These are fixed control points. For the purpose of global motion estimation it is sufficient that five of the fixed control points be detected. But if fewer than five fixed control points are detected, then of the control points detected, the ones with the smallest magnitude of the action vector are assumed as fixed control points.

Let $\gamma_i(x, y, z)$ be the coordinates of the vertices in the model corresponding to the image $I_i(x, y)$, appended with the coordinates of the search regions—i.e., $v(x', y', z, \text{texture})$. The three coordinate values for five fixed control points $P(x_i, y_i, z_i)$ are known, as the position of the splines is known and so is the corresponding position on the model, which essentially adds the value of the depth at each point on the spline.

The initial direction of spline movement is predicted in the image $I_{i+1}(x, y)$, by using the action vector direction, within the search region. If the feature is not detected in this direction then the spline is moved in the opposite direction and then in the perpendicular directions. The properties of the spline will cause the spline to find the feature edge if it can be found in whole, and if it fails to find the feature in each of these directions then the feature is partly or wholly hidden. A spline corresponding to another feature is now initialized and the fixed control points are detected.

A small number $\Delta z$ corresponds to an initial guess on the change along the $z$ axis of the detected fixed control points in $I_{i+1}(x, y)$. The fixed control points corresponding to those detected in $I_{i+1}(x, y)$ need not have all been detected in $I_i(x, y)$. This is obvious, since the coordinates of these points may have been computed from the images $I_{i+1}(x, y)$ and $I_i(x, y)$ and also since these points undergo no local motion. We now have five fixed control points $P(x_{i+1}, y_{i+1}, z_i + \Delta z)$ of $I_{i+1}(x, y)$ which correspond to $P(x_i, y_i, z_i)$ of $I_i(x, y)$.

Solving for the global estimation is done in an iterative fashion involving two recursive steps: (*i*) determination of motion parameters using given depth values and (*ii*) determination of depth values using given motion parameters.

For a detailed description on the computation involved in computing $\gamma_{i+1}(x, y, z)$ see ref. 6. The result of this process is that the vertices of the model and the search regions corresponding to the image $I_{i+1}(x, y)$ are known and the matrix transformation $T_i$ of $\gamma_{i+1}(x, y, z, 1) = T_i\gamma_i(x, y, z, 1)$ is also known for all the vertices, but the $x$ and $y$ coordinates are replaced with the detected coordinates for the detected control points. We now have $\gamma_{i+1}(x', y', z)$.

The matrix $T_i^{-1}$ is computed and $\gamma_i'(x', y', z) = T_i^{-1}\gamma_{i+1}(x', y', z)$ is computed. The two-dimensional local motion is now easily computed as the displacement between the detected nonfixed control points of $I_{i+1}(x, y)$ in $\gamma_i'(x', y', z)$ and $\gamma_i(x, y, z)$.

## V. Results of Analysis

The method for facial motion analysis outlined above is very successful, with highly accurate results. The global motion parameters were estimated with a maximum error of 25%. The local motion was also estimated with a high degree of accuracy; the larger the number of control points detected, the better the accuracy. The results were not very encouraging when fewer than five fixed control points were detected and when the splines were unable to detect the features. This resulted in the errors accumulating and finally leading to large errors and degeneration of the model. Careful and accurate mapping of the image onto the model is very important to obtain accurate results. The computed motion parameters were used to synthesize the analyzed motion.

Significant aspects of this methodology are the successful use of the action vectors to predict the local motion and the tracking of features in three dimensions, features which may be hidden to the camera in the image sequence. The results compare very well with the research done in the past.

## VI. Discussion

Model-based coding is in its infancy and many problems remain to be solved. The following are, in our opinion, some of the difficult issues in model-based coding.

**Modeling of Objects.** Modeling of objects is the most important issue in model-based coding. Complexity of analysis and synthesis depends on the model adopted.

Currently, well-approximated three-dimensional models which are obtained from *a priori* knowledge are used by most researchers. A difficult problem is how to deal with unknown objects. It seems necessary that a hierarchy of models in terms of approximation degree should be used. In this hierarchical modeling, every object is roughly modeled, and *a priori* three-dimensional models are further applied to known objects. Combination of model-based coding and waveform coding (3–5), which is proposed as a solution for the problem of unknown object, is considered as a kind of bi-level hierarchical model.

Another problem is caused by the complexity of the well-approximated model. If a model has finer details, it is more realistic for synthesis but more complex for analysis. It doesn't seem wise to use a complex model for every object. Some objects such as faces would better be modeled by using a well-approximated complex model, whereas other objects would better be coded based on imprecise models. Model hierarchy will be helpful for this problem also.

**Evaluation Problem.** For conventional waveform coding techniques, a common evaluation measure is the mean squared error. Though it has often been claimed that it is not always a good criterion, it has been a force which guides the progress of waveform coding in the right way. What is a good quality criterion for model-based coding? There has been no discussion on this point. Though Pearson (15) deals with differences between an original image and an image synthesized by texture-mapped model, squared error is still used for the quality measure. One of the attractive points of model-based coding is that it is free from squared error measure. There is no guidance at the present time on how to quantitatively evaluate the goodness of model-based coding systems.

**Promising Applications.** Currently, research is being done on model-based coding without seriously considering what is promising applications might be. Invoking this question is necessary for deciding the future direction of model-based coding. At the beginning, it was thought that model-based coding would be used for image communication. However, when we think of its high asymmetry (analysis is far more difficult than synthesis) and the growing bandwidth in future communication systems, image compression for real-time telecommunication will not be a good application for the model-based coding.

Instead, one-way-communication-type applications may be important application areas, in which database applications, broadcasting-type communication applications, and machine interface applications are included. The major advantage of model-based coding is not in compression, but in describing scenes in a structural way into codes which can be easily operated on and edited. Thus model-based coding can be applied to creating new image sequences by modeling and analyzing stored old image sequences. Such manipulations of image content may be the most important application of model-based coding.

This research was supported in part by National Science Foundation Grant IRI-89-08255 and in part by a grant from Sumitomo Electric Industries, Japan.

1. Ekman, P. & Friesen, W. V. (1977) in *Facial Action Coding System* (Consulting Psychologists Press, Palo Alto, CA).
2. Kaneko, M., Koike, A. & Hatori, Y. (1991) *Picture Coding Symp. 91*.
3. Minami, T., So, I., Mizuno, T. & Nakamura, O. (1990) *Picture Coding Symp. 90*.
4. Nakaya, Y., Aizawa, K. & Harashima, H. (1990) *Picture Coding Symp. 90*.
5. Fukuhara, T., Asai, K. & Murakami, T. (1990) *Picture Coding Symp. 90*.
6. Aizawa, K., Harashima, H. & Saito, T. (1989) *Signal Processing: Image Commun.* **1**, 139–152.
7. Choi, C. S., Aizawa, K., Harashima, H. & Takebe, T. (1990) *Proc. PCS-90* **9.15**, 1–2.
8. Forchheimer, R. (1987) *Proc. PCS-87*, 171–172.
9. Forchheimer, R. & Kronander, T. (1989) *IEEE Trans. ASSP* **37**, 2008–2023.
10. Kaneko, M., Koike, A. & Hatori, Y. (1987) *PCS-87*, **12.3.**
11. Kaneko, M., Koike, A. & Hatori, Y. (1990) *Proc. PCS-90* **9.5**, 1–2.
12. Keppei, F. & Liedtke, C.-E. (1987) *Proc. SPIE* **860**, 126–132.
13. Kass, M., Terzopoulos, A., *et al.* (1987) *ICCV* **1**, 259–267.
14. Rydfalk, M. (1989) *CANDIDE: A Parameterized Face* (Linköping University, Linköping, Sweden).
15. Pearson, D. E. (1990) *Image Commun.* **2.4**, 377–396.