FACIAL PARAMETER EXTRACTION SYSTEM BASED ON ACTIVE CONTOURS

Montse Pardàs, Marcos Losada

Universitat Politècnica de Catalunya, Barcelona, Spain

ABSTRACT

This paper addresses the application of active contours or snakes for location and tracking of facial features. Conventional snake approaches find the position of the snake by finding a minimum of its energy, composed of internal and external forces. The external forces pull the contours toward features such as lines and edges. However, in many applications this minimization leads to contours that do not represent correctly the feature we are looking for. We propose in this paper to introduce some higher level information by a statistical characterization of the snaxels that should represent the contour. This higher level information is introduced in the selection of candidates in a dynamic programming implementation of the active contours algorithm, as well as in the external energy. Furthermore, the same approach is used for tracking the contours using in this case motion estimation.

1. INTRODUCTION

A lot of attention in the past years has been devoted to contour detection and tracking by means of snake models. Snakes were first introduced by Kass et al. [5]. They proposed energy minimization as a framework where low-level information (such as image gradient or image intensity) can be combined with higher-level information (such as shape, continuity of the contour or user interactivity). In their original work the energy minimization problem was solved using a variational technique. In [1] Amini et al. proposed Dynamic Programming (DP) as a different solution to the minimization problem.

In [9], we proposed an active contours algorithm based on DP, for tracking facial features, introducing a new term in the energy of the snake, and selecting the candidate pixels for the contour (snaxels) using motion estimation. Now, this DP approach is extended to be able to automatically initialise the facial contours. In this case, the candidate snaxels are selected by a statistical characterization of the contour based on Principal Component Analysis (PCA), using what we call eigen-snaxels.

2. SNAKES

In the discrete formulation of active contour models the contour is represented as a set of snaxels $v_i=(x_i,y_i)$ for i=0,…,N-1, where x_i and y_i are the x and y coordinates of the snaxel i, and its energy, which is going to be minimized, is defined by:

$$E_{snake}(v)=\sum_{i=0}^{N-1}(E_{\text{int}}(v_i)+E_{ext}(v_i)) \tag{1}$$

We can use a discrete approximation of the second derivative to compute E_{int}.

$$E_{\text{int}}=|v_{ss}(s)|\approx|v_{i-1}-2v_i+v_{i+1}| \tag{2}$$

This is an approximation to the curvature of the contour at snaxel i, if the snaxels are equidistant. Minimizing this energy will produce smooth curves. This is only appropriate for snaxels that are not corners of the contour. More appropriated definitions for the Energy will be proposed in Section 4.2 for the initialisation of the snake and in Section 5.2 for tracking.

The purpose of the term E_{ext} is to attract the snake to desired feature points or contours in the image. In this work we have used the gradient of the image $(I(x,y))$, along the contour from v_i to v_{i+1}. Thus, the E_{ext} at snaxel v_i will depend only on the position of the snaxels v_i and v_{i+1}. That is,

$$E_{ext}(v_i)=E_{contv_i-v_{i+1}}=f(I,v_i,v_{i+1}) \tag{3}$$

However, a new term will also be added to the external energy, in order to be able to track contours that have stronger edges nearby.

3. DYNAMIC PROGRAMMING

3.1. Energy minimization

We will use the DP approach to minimize the energy in Eq. (1). Let us express the Energy of the snake remarking the dependencies of its terms:

$$E_{snake}(v)=\sum_{i=0}^{N-1}E_{\text{int}}(v_{i-1},v_i,v_{i+1})+E_{ext}(v_i,v_{i+1})=\sum_{i=0}^{N-1}E_i(v_{i-1},v_i,v_{i+1}) \tag{4}$$

Although snakes can be open or closed, the DP approach can be applied directly only to open snakes. In this case the limits of Eq. 4 are adjusted to 1 and N-2 respectively.

Now, as described in [1], this energy can be minimized via discrete DP defining a two-element vector of state variables in the ith decision stage: (v_{i+1}, v_i). The optimal value function is a function of two adjacent points on the contour $S_i(v_{i+1},v_i)$, and can be calculated, for every couple of possible positions for snaxels $v_{i+1}$and v_i, as:

$$S_i(v_{i+1},v_i) = \min_{v_{i-1}}\left[S_{i-1}(v_i,v_{i-1}) + E_i(v_{i-1},v_i,v_{i+1})\right] \qquad (5)$$

$S_0(v1,\ v0)$ is initialised to $E_{ext}(v0,v1)$ for every possible candidate pair $(v0,v1)$ and from this, S_i can be computed iteratively from $i=1$ up to $i=N-2$ for every candidate position for v_i. The total energy of the snake will be

$$E_{snake}(v) = \min_{v_{N-1}} S_{N-2}(v_{N-1},v_{N-2}) \qquad (6)$$

Besides, we have to store at every step i a matrix which stores the position of v_{i-1}that minimizes Eq. (5), that is,

$M_i(v_{i+1},v_i) = v_{i-1}$ such that v_{i-1}minimizes (5).

By backtracking from the final energy of the snake and using matrix M_i, the optimal position for every snaxel can be found.

In the case of a closed contour the solution proposed in [2] is to impose the first and last snaxels to be the same, and fix it to a given candidate for this position. The application of the DP algorithm will produce the best result under this restriction. Then this initial and final snaxel is successively changed to all the possible candidates, and the one that produces a smaller energy will be selected. We use an approximation proposed in [3] that requires only two open contour optimisation steps.

3.2. Selection of candidates

Up to now, we have assumed that for every snaxel v_i there are a finite (and hopefully small) number of candidates, but we have omitted how to select these candidates. The computational complexity of each optimisation step is O(nm^3), where n is the number of snaxels and m the number of candidates for every snaxel. Thus, it is very important to maintain m low.

In [1] only a small neighbourhood around the previous position of the snaxel was considered. However, the algorithm was iteratively applied starting from the obtained solution until there was no change in the total energy of the snake. This method has several disadvantages. First, like in the approaches which use variational techniques for the minimization, the snake can fall into a local minimum. Second, the computational time can be very high if the initialisation is far from the minimum.

In [2] and [3] a different set of candidates is considered for every snaxel. In particular, [2] establishes uncertainty lists for the high curvature points and defines a search space between these uncertainty lists. In [3] the search zone is defined with two initial concentric contours. Each contour point is constrained to lie on a line joining these two initial contours. This approach gives very good results if the two concentric contours that contain the expected contour are available and the contour being tracked is the absolute minima in this area. However, these concentric contours are not always available.

In the next sections we will describe how we can select these candidates for facial feature point detection and tracking, respectively.

4. FACIAL FEATURE POINT DETECTION

4.1. Selection of candidates

We propose a new method that in a first step needs to fix the topology of the snakes. In our case, we are using a 16 snaxels equally spaced snake for the mouth and a 8 snaxels equally spaced snake for the eyebrows. To select the best candidates for each of these snaxels we compute what we call the v_i-eigen-snaxel, by extracting samples of them from a database. That is, after resizing the faces from our database to the same size (200x350), we extract for each snaxel v_i the 16x16 area around the snaxel in every image of the database. The extracted sub-images are used to form the training set of vectors for the snaxel v_i, and from them, the eigenvectors (eigen-snaxels) are computed by classical PCA techniques.

The first step for initialising the snakes in a new image is to roughly locate the face. Different techniques can be used for this aim [6], [10]. After size normalization of the face area, a large area around the rough position for every snaxel v_i is examined by computing the reconstruction error with the corresponding v_i-eigensnaxel. Those pixels leading to the smaller reconstruction error are considered as candidates for the snaxel v_i.

4.1.1. Principal Component Analysis (PCA)

The principle of PCA is to characterise a set of very similar images by a few variables reducing the dimension of the set [8].

For each snaxel v_i, the vectors $\{x^i\}$ are constructed by lexicographic ordering of the pixel elements of each sub-image from the training set. A partial KLT is performed on these vectors to identify the largest-eigenvalue eigenvectors.

4.1.2. Distance measure

To evaluate the feasibility for a given pixel to be a given snaxel we first construct the vector $\{x^i\}$ using the corresponding sub-image. Then we obtain a principal component feature vector $y=\phi_M^T\,\tilde{x}$, where $\tilde{x} = x - \bar{x}$ is the mean-normalized image vector, ϕ is the eigenvector matrix for snaxel v_i and ϕ_M is a sub-matrix of ϕ that contains the M principal eigenvectors. The reconstruction error is calculated as:

$$\varepsilon^2(x) = \sum_{i=M+1}^{N} y_i^2 = \|\tilde{x}\|^2 - \sum_{i=1}^{M} y_i^2 \qquad (7)$$

The pixels producing the smaller reconstruction error are selected as candidate locations for the snaxel v_i.

4.2. Snake energy

4.2.1. External energy

A new term is added to the external Energy function in Eq (3) to take into account the result obtained in the PCA process. The energy will be lower in those positions where the reconstruction error is lower.

$$E_{ext}(v_i) = \gamma E_{contv_{i-1}-v_i} + (1-\gamma)\varepsilon^2(v_i) \qquad (8)$$

In our experiments the value of γ has been set to 0.5.

4.2.2. Internal energy

As mentioned, the Energy (2) is only appropriate for snaxels that are not corners of the contour. In the case of corners the energy has to be low when the second derivative is high. We use, for the energy of the snaxels belonging to the eyebrows and the mouth:

$$E_{\text{int}}(v_i) = \left[\beta_i \left|v_{i-1} - 2v_i + v_{i+1}\right| + (1-\beta_i)\left(B - \left|v_{i-1} - 2v_i + v_{i+1}\right|\right)\right] \qquad (9)$$

where β_i is set to 1 if v_i is not a corner point and to 0 if it is. B represents the maximum value that the approximation of the second derivative can take.

To solve problems of snaxel grouping we also add, as in [7], another term to the internal energy that forces snaxels to preserve the distance between them from one frame to the next one:

$$E_{dist_i} = \left| u_i^{t+1} - u_i^t \right| + \left| u_{i+1}^{t+1} - u_{i+1}^t \right| \qquad (10)$$

$u_i^t = v_i - v_{i-1}$, in the frame t

So when the distance is altered the energy increases proportionally.

5. FEATURE POINT TRACKING

5.1. Selection of candidates

In the initialisation of the snake, the candidates for every snaxel were selected on the basis of a statistical characterization of the texture around them. However, in the case of tracking, we have a better knowledge of this texture if we use the previous frame. The solution we propose to find the candidates for every snaxel in the tracking process uses motion estimation in order to select the search space for every snaxel.

A small region around every snaxel is selected as basis for the motion estimation. The shape of this region is rectangular and its size is application dependent. However, the region should be small enough so that its motion can be approximated by a translational motion. The compensation error for all the possible displacements (dx,dy) of the block in a given range is computed as:

$$MCE_{vi0}(dx,dy) = \sum_{j=-Ry}^{j=Ry} \sum_{i=-Rx}^{i=Rx} \left| I_{t-1}(x_0 - i, y_0 - j) - I_t(x_0 - i + dx, y_0 - j + dy) \right|^2 \qquad (11)$$

being *(x_0, y_0)* the x and y coordinates of the snaxel v_i in the previous frame, which we have called v_{i0}. The region under consideration is centred at the snaxel and with size *2Rx* in the horizontal dimension and *2Ry* in the vertical dimension. The range for *(dx,dy)* determines the maximum displacement that a snaxel can suffer. The matrix *$MCE_{vi0}(dx,dy)$* is stored for every snaxel, and the M best results are selected as possible new locations for snaxel v_i.

5.2. Snake energy

5.2.1. External energy

We use for the external energy in the tracking procedure the same principle than for the initialisation. That is, it will be composed, as in (8), of two terms. The first one is the gradient along the contour, but the second one is slightly different, as in this case we can use the texture provided by the previous frame instead of the statistical characterization. Thus, the second term corresponds to the compensation error obtained in the motion estimation. In this way preference is given to those positions with the smaller compensation error. That is, the energy will be lower in those positions which texture is most similar to the texture around the position of the corresponding snaxel in the previous frame. Therefore, the expression for the external energy will be:

$$E_{ext}(v_i) = \gamma E_{contv_{i-1}-v_i} + (1-\gamma) MCE_{v_{i0}}(v_i) \qquad (12)$$

The constant γ can be set depending on the strength of the contour that is being tracked. If it is a strong contour γ is chosen close to 1. Otherwise, more importance is given to the Motion Compensation Error term.

5.2.2. Internal energy

The internal energy we use to track feature points has the same formulation as in paragraph 4.2.2. We assume here that we know the geometry of the different face features to correctly set β_i to 1 or 0, Eq. (9).

The same algorithm can be used to track generic objects, but in this case we have to modify the way we calculate the internal energy in order not to have to specify the β_i parameter value. Thus, for a generic object, instead of (9), we use:

$$E_{\text{int}}(v_i) = \left| C_i^t - C_i^{t+1} \right| \qquad (13)$$

$$C_i^t = \left| \hat{u}_{i+1}^t - \hat{u}_i^t \right|^2$$

$u_i^t = v_i - v_{i-1}$, in the frame t

$\hat{u}_{i+1}^t$ y $\hat{u}_i^t$ are unit vectors. As described in [7], this internal energy tends to keep the curvature of the snake between two successive frames.

6. RESULTS AND CONCLUSIONS

In Figure 1 and 2 we show some examples of automatic initialisation and tracking. More results will be shown in our web page by the time of the conference. To perform the tests, the algorithms have been introduced in a GUI that allows manual correction of the snaxels position. In the tests performed, the contours obtained with the automatic initialisation always correspond to the facial features, if the face has been located accurately. However, as shown in Figure 2, sometimes some points should be manually modified if more accuracy is required. The tracking has been performed on 136 sequences from the Cohn-Kanade facial expression database [4]. From these sequences, 92 needed no manual correction, 24 needed 1, 2 or 3 points correction along the sequence and 20 sequences had one major problem in at least one frame for one feature tracking.

We have proposed in this paper a criterion for selection of the candidate snaxels in a dynamic approach implementation of the active contours algorithm. We also claim that, in order to initialise or track generic contours, the external energy needs an additional term related to the specific texture around the snaxels, in order to avoid the contour to fall in stronger edges that might be around the one we are aiming.

7. REFERENCES

[1] A. Amini, T. Weymouth and R. Jain, "Using Dynamic Programming for Solving Variational Problems in Vision", IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 12, No. 9, September 1990.

[2] D. Geiger, A. Gupta, L. Costa and J. Vlontzos, "Dynamic Programming for Detection, Tracking, and Matching Deformable Contours", IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 17, No. 3, March 1995.

[3] S. Gunn and M. Nixon, "Global and Local Active Contours for Head Boundary Extraction", International Journal of Computer Vision 30(1), pp. 43-54, 1998.

[4] Kanade, T., Cohn, J.F., Tian. Y., Comprehensive Database for Facial Expression Analysis, Proceedings of the Fourth IEEE International Conference on Automatic Face and Gestures Recognition, Grenoble, France, 2000.

[5] M. Kass, A. Witkin and D. Terzopoulos, "Snakes: Active Contour Models", International Journal of Computer Vision, Vol. 1, No. 4, pp. 321-331, 1988.

[6] C. Kervrann, F. Davoine, P. Perez, R. Forchheimer and C. Labit, "Generalized likelihood ratio-based face detection and extraction of mouth features", Pattern Recognition letters 18, pp. 899-912, 1997.

[7] Chun Leung Lam, Shiu Yin Yuen, "An unbiased active contour algorithm for object tracking", Pattern Recognition Letters 19, pp. 491-498, 1998.

[8] B. Moghaddam, A. Pentland, "Probabilistic Visual Learning for Object Detection", in 5th International Conference on Computer Vision, Cambridge, MA, June 95.

[9] M. Pardàs, E. Sayrol, "A new approach to active contours for tracking" in Proceedings of Int. Conf on Image Processing, ICIP 2000, Vancouver, Canada, September 2000.

[10] V. Vilaplana, F. Marqués, P. Salembier, L Garrido, "Region Based Segmentation and Tracking of Human Faces", Proceedings of European Signal Processing Conference (EUSIPCO-98), pp. 311-314, 1998.

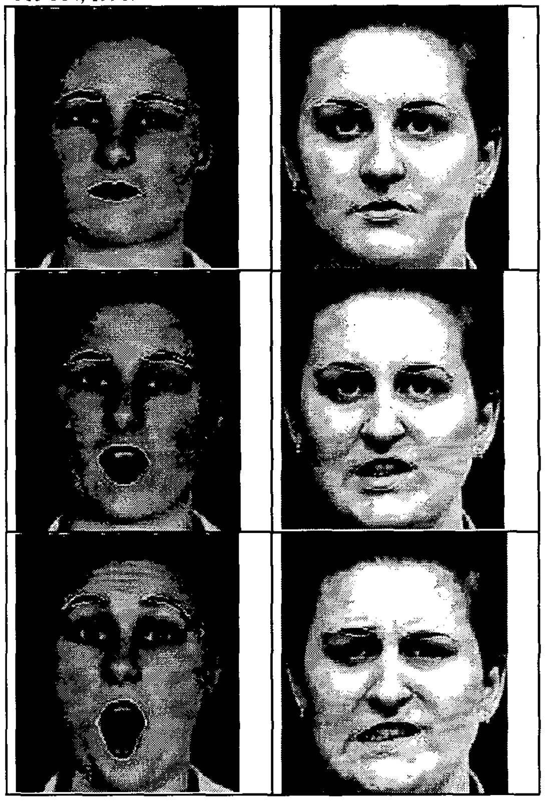
Fig. 1. The first line are examples of automatic initialisation of the face features, while the second and third lines show tracking results.

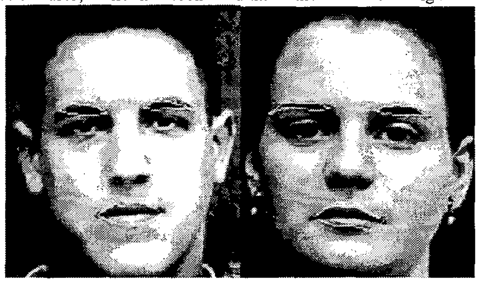
Fig. 2. Automatic initialisation examples with minor errors