Robust Facial Feature Tracking using Selected Multi-Resolution Linear Predictors

Eng-Jon Ong
CVSSP,
University of Surrey, UK

Yuxuan Lan, Barry Theobald,
Richard Harvey
SCS, University of East Anglia, UK

Richard Bowden
CVSSP,
University of Surrey, UK

Abstract

This paper proposes a learnt data-driven *approach for accurate, real-time tracking of facial features using only intensity information. Constraints such as* a-priori *shape models or temporal models for dynamics are not required or used. Tracking facial features simply becomes the independent tracking of a set of points on the face. This allows us to cope with facial configurations not present in the training data. Tracking is achieved via linear predictors which provide a fast and effective method for mapping pixel-level information to tracked feature position displacements. To improve on this, a novel and robust* biased *linear predictor is proposed in this paper. Multiple linear predictors are grouped into a rigid flock to increase robustness. To further improve tracking accuracy, a novel probabilistic selection method is used to identify relevant visual areas for tracking a feature point. These selected flocks are then combined into a hierarchical multi-resolution LP model. Experimental results also show that this method performs more robustly and accurately than AAMs, without any* a priori *shape information and with minimal training examples.*

1. Introduction

In this paper, we propose a learnt person-specific but importantly, a *data-driven* approach to achieve accurate and real-time tracking of facial features using only intensity information. Here, additional constraints such as *a-priori* shape models or temporal models of dynamics are neither required nor used. Through the accurate tracking of independent features, it is possible to deal with configurations of facial features that were not present in the training data. There exists a number of different methods for facial feature tracking. One class of popular methods is the model-based approach using active contours. An example of this is used for tracking lip contours [2, 13]. This was improved in [1] by coupling this technique with 2D templates. In [12], temporal constraints were also included to improve on tracking. Specific parts of the face (e.g. lips) can be initially segmented using colour and markov random field models before obtaining the final shape using active contours [7]. Other methods include active appearance models (AAM) [3] for tracking lip shapes [8], where relevant training data is obtained from audio [4].

The proposed method in this paper builds upon the work in [9]. Here, to track features, the method of linear predictor flocks was used. Each linear predictor provides a mapping from sparse template differences to the displacement vector of a tracked facial feature. Multiple LPs can then be grouped into rigid flocks to track a single feature point with greater robustness and accuracy. Considerable novelty is added to [9] to produce a state of the art facial feature tracking framework as follows: Firstly, we extend the domain of tracking lip shapes to that of general facial feature tracking. Secondly, we propose a novel type of linear predictors known as biased-LPs (Section 2). Thirdly, a novel LP selection method based on probabilistic selection is proposed, removing the need for a heuristically defined threshold for how many LPs to retain in a flock. Instead, the proposed selection method migrates member LPs into optimal positions (Section 4). Fourthly, to further increase tracking robustness, a hierarchical multi-resolution approach is used to combine together LP flocks of different sizes (Section 5). Furthermore, we thoroughly quantify and compare the robustness of the proposed method at different parameter settings in Section 6. A comparison is also made with AAMs which employ heavy dependancies between features in the form of shape priors. We also demonstrate the use of the proposed methods for accurately tracking various facial features before concluding in Section 7.

2. Biased Linear Predictors

A Linear Predictor (LP) forms the central component of the proposed tracking mechanism. An LP is responsible for tracking a particular visual feature by means of a linear mapping from an input space of sparse support pixels to a displacement vector space, the motion of the feature

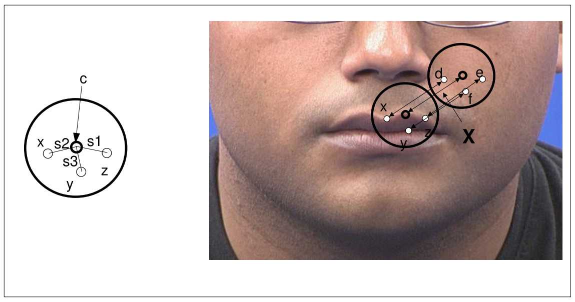

Figure 1. Illustration of a linear predictor(LP). Each LP has a reference point c. Within an area around c, called the support region a set of randomly sampled support pixels $(x, y, z)$with their offsets from c: $s1, s2, s3$. Also shown is the synthesis of training data. X is the artificial translation of c. The corresponding support pixel difference vector is $\delta p = (x, y, z)$ - (d, e, f).

point. Recently, linear predictors have been used for efficient constrained tracking of planar objects[14]. Along similar lines, Relevance Vector Machines (RVMs) were used to provide displacement predictions [11]. More recently, Bayesian Mixtures of Experts coupled with RVMs have been proposed for more accurate tracking [10]. However, compared with LPs, both the above approaches are higher in complexity.

A linear predictor (LP) is defined as a set of five components, $L = \{c, H, V, S, b\}$, where c is a 2D reference point defining the location of the feature, H is the linear mapping to a displacement vector, S is a set of 2D offsets for positioning the support pixels, $|S|$ is the number of support pixels, V is a $|S|$-dimensional vector of base support pixel values. V forms a sparse template for the visual appearance of the feature point. Additionally, we improve on the original work on LPs by adding a bias factor b to the linear mapping H. As can be seen in Section 6.2, this simple but effective addition significantly increases the tracking accuracy. From here, we define this new type of linear predictors as *biased LPs*.

In order to obtain V, $S = {s_i}_{i=1}^{|S|}$ is defined and used, where s_i is the offset relative to c. The offset positions s_i are obtained as random offsets within a specified radius from the origin. In Section 5, we show how LPs of different sizes can be combined together into multi resolution LPs for much better robustness. An illustration of a linear predictor can be seen in Figure 1. To use a biased LP to predict the displacement of its tracked feature (i.e. reference point c), given the image $I = I_{ij}^{H,W}$ of dimensions HxW as input, we firstly obtain the difference between the base support pixel values and those from the current image:

$$\delta p = (V_i - V_i^I)_{i=1}^{|S|} \tag{1}$$

where $V_i^I = I_{c+s_i}$ is the pixel value at position $c + s_i$ in the image. The biased displacement of c, t is then:

$$t = H\delta p + b \tag{2}$$

2.1. Learning the Biased Linear Mapping

The linear mapping (H) of an LP is learnt using least squares optimisation. As a result, from Eq. 2, we need a set of training data in the form of support pixel differences (δp) and displacement vector pairs (t). To achieve this, a number of training examples can be synthesised from each single training image.

It is assumed that in each training image, the location of the tracked feature point is ground-truthed, which is also the value of the LP reference point (c). This allows us to extract the base support pixel values (V). Following this, it is possible to synthesise a number of random displacements from c. Along with these displacements, we can also obtain their respective δp vectors by initially translating c by the displacement, obtaining the support pixel values at that position and calculating its difference from the base support pixel values. This process is then repeated for all training images, allowing us to gather a wide range of training examples for learning the linear map H. The base support pixel values are then set as their respective mean intensities across all training images.

The generated examples can then be compiled into the following matrices: T and δP, where T is a 2xN_T matrix, of which each column is a displacement vector. Similarly, δP is a $|S|$xN_T matrix, where each column is the displacement vector's corresponding support pixel difference vector. Additionally, we learn the bias for the linear model by adding an additional column of 1s to the end of δP, giving: $\delta P' = (\delta P, [1])$, where $[1]$ denotes a column vector of rows N_T. Using least squares, H can now be obtained as follows:

$$H = T\delta P^{'+} = T\delta P^T(\delta P\delta P^{'T})^{-1} \tag{3}$$

where $\delta P^{'+}$ is the pseudo-inverse of $\delta P'$.

3. Rigid Flocks of Biased LPs

Using a single linear predictor to determine the displacement of a feature point is insufficient. This is because a single linear mapping between the support pixel difference values to the displacement space is seldom robust to noise, illumination changes and other image warps that may occur on the feature point and its surroundings. This problem can be addressed by grouping multiple linear predictors together into a *rigid flock* of LPs. In previous work [6, 5], a flock tends to be a loose collection of features or trackers that must lie within an area surrounding a reference point (e.g. feature mean position). Whilst each flock member may move somewhat independently within this area, in general they agree on the general direction of the tracked target.

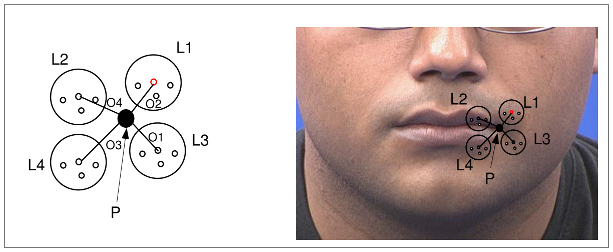

Figure 2. Illustration of a rigid flock of linear predictors, whose position is given by reference point, P. The member LPs are $(L1, L2, L3, L4)$ each with a rigid offset from P: $O1, O2, O3, O4$.

This agreement often cancels out noise present in the individual tracker predictions. In our case of a rigid flock, the LP trackers are always fixed to the original offset away from the reference point. In order to further improve on the robustness and accuracy of the tracking predictions, separate flocks are used for predicting the individual x and y displacement components. To disambiguate between these two types of flocks, the superscript x and y are used respectively. Since both of these LP flocks are similar in form, for the following LP flock definitions, we use a "*" to act as a placeholder that can take either the superscript x or y. This removes the need to repeat the formal definitions twice.

Thus, a rigid flock of LPs consists of the following components: a reference point P^{F*}, a set containing L^{F*} number of linear predictors ($L^{F*} = \{L^*\}_{f=1}^{|L^{F*}|}$) and the 2x$|L^{F*}|$ matrix of linear predictor offsets (O^{F*}) from P^{F*} (Figure 2). We define the displacement predictions obtained from Eq. 2, of each of the member linear sets as $\{t^*\}_{f=1}^{|L^{F*}|}$. This arrangement allows us to have a reference point offset from the centre of the LPs in the flock, but still be guided by its predictions. This is in contrast to, for example, taking the reference point as the mean of the member LPs, where it is forced to lie in the centre of the flock members. As we will see in Section 6, a rigid flock of LPs combined with carefully selected LPs is critical to increasing the tracking accuracy and fundamentally different to related approaches (e.g [6, 5, 3, 1]).

The 2D image displacement prediction of a feature point modeled with 2 LP flocks (L^{Fx}, L^{Fy}) is given as:

$$x^F = (1/|L^{Fx}|) \sum_{f=1}^{|L^{Fx}|} t_f^x(1) \quad (4)$$

$$y^F = (1/|L^{Fy}|) \sum_{f=1}^{|L^{Fy}|} t_f^y(2) \quad (5)$$

where $t_i^*(1)$ and $t_i^*(2)$ are respectively the x and y components of the individual LP predicted displacement vectors. This is then used to update the position of the rigid flock reference point for both x and y LP flocks: $P^{Fx} = P^{Fx} + (x^F, y^F)$ and $P^{Fy} = P^{Fy} + (x^F, y^F)$.

4. Probabilistic Selection of LPs

Having defined a rigid flock of LPs in the previous section, we are now faced with the crucial issue of deciding *where* to place the member LPs. To start, for both sets of LP flocks (L^{Fx}, L^{Fy}), a predetermined number of LPs are randomly scattered within an area around the rigid flocks' reference point. It is then possible, with all these linear predictors to use Eq. 4 and 5 to predict the displacement of the reference point, given a new input image.

However, this can be suboptimal, since there may exist many LPs in a flock that will give wrong displacement predictions. We now are faced with the crucial problem of identifying meaningful context useful for tracking a particular feature point. To address this issue, this section proposes an iterative and probabilistic method for selecting *separate* sets of LPs for accurate and robust predictions. More specifically, this method will be based on iteratively selecting new sets of LPs based on their displacement prediction mean errors from training groundtruth data. In earlier work [9], a naive method of removing LPs in the rigid flock with mean prediction errors less than a predefined threshold was used. Here, with an iterative scheme, a threshold is no longer needed.

To continue, a number of definitions is firstly given: The training set will be a set of N_G images I^G with groundtruth positions for the target feature. The displacement groundtruth dataset is defined as $G = (g_{x,t}, g_{y,t})_{t=1}^{N_G}$, where each example, g_i, is a 2D displacement vector. Given a rigid flock learnt using a small number of training examples, it is possible to track the feature using Eq. 4 and 5 and obtain the predicted displacement vectors for every LP at every frame, which is defined as $(x'^F_{i,t}, y'^F_{i,t})_{i=1}^{N_G}$.

4.1. Probabilistic Re-selection using Individual LP Mean Error

The biased LP selection method proposed here is similar to the first step of particle selection used in particle filters based trackers. The proposed method is essentially an iterative method, where at each iteration, we draw a new set of location offsets for member LPs in a flock with respect to weightings based on an inverted form of training error. Following this, it is possible to relearn the linear mappings for this new set of LPs and reiterate the whole process to progressively locate better locations for placing LPs resulting in greater tracking accuracy.

To start, we calculate the two displacement prediction component's mean error for each LP in a rigid flock for it-

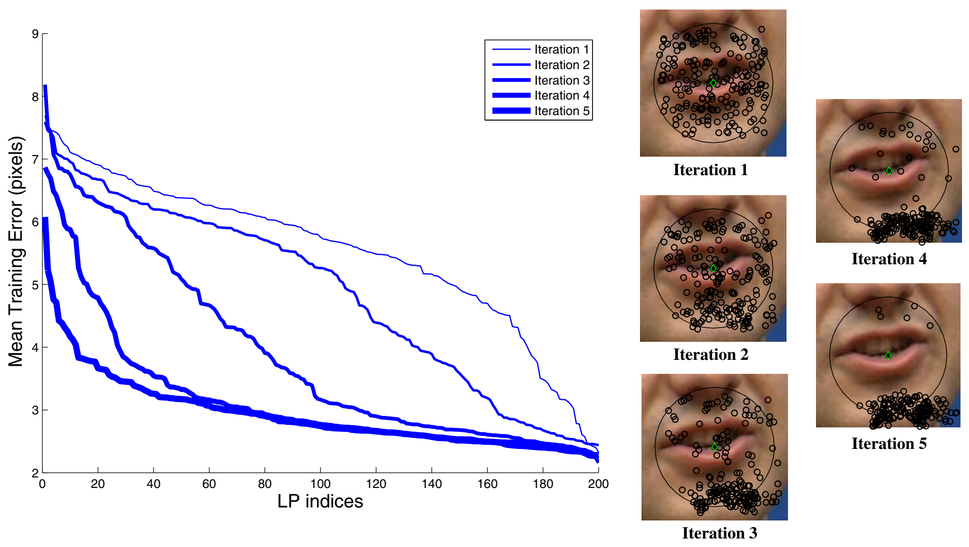

Figure 3. Shown are the sorted training errors of individual LPs in a flock over different iterations of the probabilistic selection. The error graph on the left shows that the errors of a majority of LPs are reduced through iterative selection. Shown on the right are the locations of the different LPs for tracking a point on the lower inner lip at different iterations of the selection process.

eration step α as follows:

$$\epsilon^{\alpha}_{x,i} = (1/|G|)\sum_{t=1}^{N_G}\sqrt{(g_{x,t}x'^{F}_{i,t})^2} \tag{6}$$

$$\epsilon^{\alpha}_{y,i} = (1/|G|)\sum_{t=1}^{N_G}\sqrt{(g_{y,t}y'^{F}_{i,t})^2} \tag{7}$$

Having obtained the individual LP mean errors, the next step is to transform them into a selection probability value. This essentially involves inverting the error values. There are a wide variety of methods to achieve this, however, we have found that given a set of errors $\epsilon^{\alpha}_{*,i}$ where $*$ can either be x or y, the errors can be inverted to selection probabilities ($\beta^{\alpha}_{*,j}{}^{M}_{j=1}$) using:

$$\beta^{\alpha}_{*,j} = \frac{\max(\epsilon^{\alpha}_{*,i}) - \epsilon^{\alpha}_{*,i}}{\sum \epsilon^{\alpha}_{*,i}{}^{M}_{i=1}} \tag{8}$$

Following this, a cumulative probability graph can be formed:

$$\gamma^{\alpha}_{*,j} = \sum \epsilon^{\alpha}_{*,i}{}^{j}_{i=1} \tag{9}$$

From this, we can probabilistically select the next batch of LPs from the existing set in an iterative manner. Here, it is possible to iterate for a fixed number of times, or until the error has converged (i.e. the change in error is less than a preset threshold T^{ϵ}). The selection algorithm is given in Algorithm 1. An illustration of how the positions of the members LPs in a flock changes as the algorithm progresses can be seen in Figure 3.

Algorithm 1 Probabilistic Selection Algorithm

```
$\alpha = 1$
$\delta\epsilon = T^{\epsilon} + 1$
Start with the initial LP flock $(L^{F*,\alpha})$ whose LP members' offset positions $(O^{*,\alpha})$ were initialised randomly around its respective feature point $P^{*}$.
while $\alpha < N^{\alpha}$ or $\delta\epsilon < T^{\epsilon}$ do
    Create a LP offset matrix, whose elements are 0: $O^{F*,\alpha}$
    for $i = [1...M]$ do
        $\alpha = \alpha + 1$
        Generate random number $R^{\alpha} \in \{0, ..., 1\}$
        Obtain index for new LP: $\arg_j \min \gamma^{\alpha}_{*,j}$
        Get the offset position of the selected LP: $(o_x, o_y) = O^{F*,\alpha-1}_{j}$
        Add a small random noise offset $(n_x, n_y)$ to it: $(o_x, o_y) += (n_x, n_y)$
        Set the position of the LP in the new set as $(o_x, o_y)$: $O^{F*,\alpha}_{i} = (o_x, o_y)$
    end for
    Relearn the linear mappings for the current iteration's LP set $(L^{F*,\alpha})$ using the newly obtained offsets $O^{F*,\alpha}$
end while
```

5. Chain of Multi-Scale Selected LP Flocks

It is known from previous work on LPs [14] that increasing the support region size and the range of training displacements result in greater robustness to large displacements from a feature's location, with reduced accuracy as a

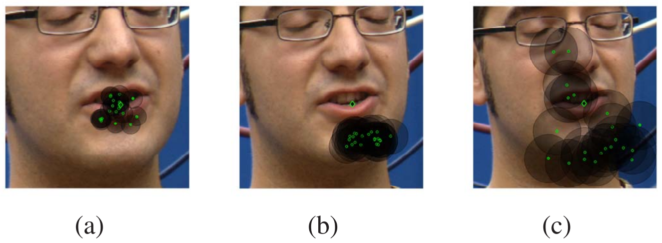
Figure 4. Shown here are the different sized LP flocks that form a multi-scale lp flock, where (a) is the smallest to (c) being the largest. Predicting the feature displacement cascades from the largest (c) to the smallest (a) LP flock.

trade off. In order to obtain tracking results that are *both* robust and accurate, a chain of LPs of decreasing sizes can be used. The largest LP is first used to predict a feature's location. The result is then passed to a less robust but more accurate LP down the chain, and repeated until we have reached the end of the LP chain, with an accurate location of a displaced feature.

We have found that the above argument also holds true for our case of selected LP flocks. As a result, we employ a similar strategy for obtaining a robust and accurate feature tracking by chaining together a sequence of decreasing size LP flocks. Here, we define the *size* of an LP flock by the radius of feature displacements that it has been trained to cope with. This also affects the offset range for member LPs as well as the support regions of the individual member LPs (see Figure 4). The exact sizes of the support regions, training displacement ranges and LP member offset radius will be given in Section 6. Formally, suppose we have trained N_S number of different sized LP flock pairs for tracking a feature point. It must be noted that the sizes of the member LPs of each pair is the same. More specifically, an LP flock pair of size θ is defined as: $L^{\theta} = (L^{Fx,\theta}, L^{Fx,\theta})$. These flocks can be sorted in descending order based on their sizes and used to form an ordered set to represent the multi-scale LP flock chain: $\{L^{\theta_i}\}_{i=1}^{N_S}, \theta_{i+1} < \theta_i$, where all the N_S flock sizes are defined as the set $\{\theta_i\}_{i=1}^{N_S}$.

We can now use the multi-resolution LP flock chain to predict a feature's position. Suppose our input image is given as I. The initial starting position for prediction is given by the reference point position (Section 3) of the largest LP flock. A prediction is then made on the location of the tracked feature using Eq. 5. The resulting feature position, given by the updated reference point P^{F*,θ_1} is used to initialise the reference position of the next flock in the chain, that is: $P^{F*,\theta_2} = P^{F*,\theta_1}$. This process is the repeated until we have reached the last scale θ_{N_S}. The updated reference position of the last LP flock $P^{F*,\theta_{N_S}}$ is used as the tracked position of the feature of interest.

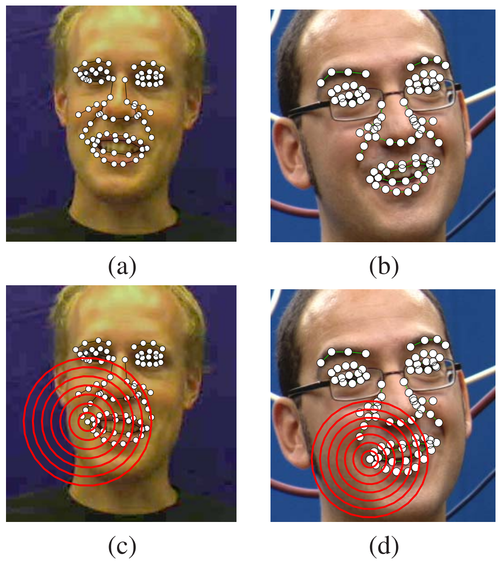
Figure 5. Shown are all 83 tracked points for the webcam data(a) and the SD camera data(b). The displacement error sizes (up to 60 pixels) with respect to the face is illustrated in (c) and (d).

6. Experiments

This section describes experiments carried out to further understand the following points: firstly how the proposed method's robustness is affected with regards to different parameters, in particular the usefulness of the bias component in an LP (Section 6.2), the total number of LPs that is used in a flock (Section 6.5) and the role of different sized LPs in a multi-scale LP flock model (Section 6.5); secondly, a comparison of the tracking performance of multi-scale LP flocks are compared to those from an AAM method [8] (Section 6.6). AAMs were chosen due to their known performance in tracking facial features without needing a large amount of training examples.

6.1. Experimental Setup

For the experiments, two classes of data were captured: with a standard definition (PAL) resolution camera; and with a standard 640x480 resolution webcam. Using the SD camera, two separate video sequences from each of 3 different subjects engaged in a natural conversation with another person were captured. Similarly, using the webcam, two separate video sequences from each of 4 different subjects discussing the contents of pictures with another person were captured. For each subject, one sequence was retained for training purposes and the remaining for testing. The test sequences contained approximately 800 to 1000 frames. In order to groundtruth the data, semi-automatic labelling was used, since hand labelling every frame would have been too time consuming. In total, 83 points on the face were tracked as shown in Figure 5a,b. In order to track all of these points, 83 independent multi-resolution LP flocks were used. For training the LPs' linear mapping matrix H, around 13-15 images extracted from the training sequence was used. It is important to note that the training size is very small in comparison to the size of the test data. Each LP had a support region of half the size of its associated scale, with 80 support pixels randomly placed within this area.

In order to quantify the robustness of the proposed method at different parameter settings and to compare against AAMs, a test measuring convergence to groundtruth from displaced data was used. Specifically, for each test image in the test sequence, all the tracked points are initially displaced a certain distance from their groundtruth positions. These displacements range from 1 to 60 pixels in radius (Figure 5c,d). A tracking method (e.g. proposed LP method or AAMs) is then applied to these corrupted positions. A count of how many points eventually fall within a certain radius of the groundtruth data is made. Here, we have set the convergence radius to 5 pixels. It is then possible to obtain the percentage of points across all test images and tracked points that have fallen within the convergence region for a given range of displacement noise.

6.2. Biased LPs vs Original LPs

In order to analyse the improvements due to the addition of a bias in the LPs, convergence tests were performed on selected LP flocks with 3 scales with and without the bias. All LP flocks have 200 members each. The results can be seen in the first row of Figure 6. We see that for the webcam data, there is a consistent improvement across the entire error range. In the SD dataset, we find that the bias provided increased robustness, especially when the noise is large.

6.3. LP Selection

To better understand how much improvement is obtained from the LP probabilistic selection method, convergence tests with the selection method switched on and switched off were performed. For these experiments, 200 LPs were used per flock, and 3 scales were used: 10,40 and 60 pixels. The results can be seen in the second row of Figure 6. Here, it can be seen for both SD and webcam quality data that the selection process results in LP trackers that are significantly more accurate and robust than those with LP flocks whose members were randomly placed.

6.4. LP Flock Size

The next set of experiments were used to determine the effect of increasing the number of members in an LP flock. To this end, tests were performed using LP flocks with the following number of members: 20,40,80,120,160,200. These flocks underwent the selection process. Additionally, 3 scales were used as before: 10, 40 and 60 pixels. The results can be seen in the third row of Figure 6. For the webcam database, it was interesting to note that increasing the flock size beyond 80 members did little to improve the robustness of the method. However, the case was different in terms of the SD dataset, whereby increasing the sizes re-

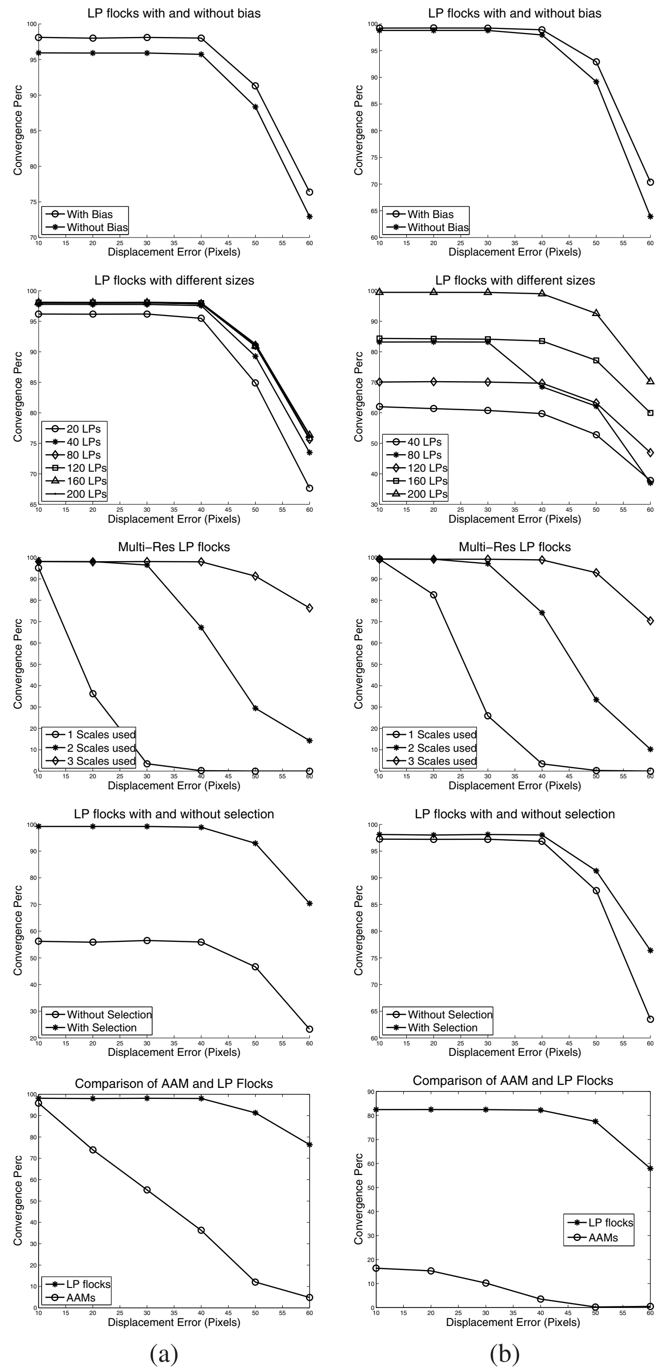

Figure 6. The convergence graphs for the different LP parameters settings for webcam sequences(a) and SD camera sequences (b). For more details, refer to Section 6.

sulted in continuous improvement in robustness.

6.5. Role of Different LP Flock Scales

In order to analyse the role of the different LP sizes in a multi-resolution LP flock, tests were performed with LP flocks with increasing number of scales. Here, a maximum of 3 scales are used, of sizes 10, 40 and 60 pixels respectively. Each individual scale has selected flocks with 200 members. Tests were firstly performed with LP flocks with LPs of size 10 pixels. Next, LP flocks of size 40 were added and the convergence tests repeated. Finally, the LPs of size 60 were added and the results compiled. The obtained convergence graphs is shown on the fourth row of Figure 6. It can be seen that adding the larger scales provide significant improvements on the convergence scores when the noise level is high (i.e. 60 pixels radius).

6.6. Comparison with AAMs

To compare the proposed method against AAMs, we trained separate AAMs for each subject using exactly the same training data used for the LP flocks. The convergence scores using the AAM method was then obtained and compared against a multi-resolution LP selected-flocks with 3 scales and 200 member LPs (last row of Figure 6). We find that the AAM performs poorly when the displacement error is high. In the webcam data, when the displacement error is small (<10 pixels), the performance of the AAM is comparable to the proposed method. However, it is surprising that the AAMs performed poorly when the image quality was higher for the SD dataset. This was despite it using both shape and texture constraints that were employed in a multi-resolution optimisation process.

6.7. Tracking Results

The results of the proposed tracking method is illustrated in Figure 7 and Figure 8. For a more complete view of the results, we direct the reader to the video sequences in the supplementary material. The tracking speed in unoptimised C++ is about 20fps on a standard single-core processor PC. Despite the facial features being tracked independently, the trackers managed to recover from tracking errors. This was done in the presence of simultaneous pose and expression changes. The webcam data required the method to be robust to low image quality due to significant compression artifacts and image noise as well as a low capture framerate of 15fps. Note that the only temporal constraint used was to initialise the LPs at every frame from the position in the previous frame. A verification of these claims can be seen in the video sequences provided in the supplementary material.

7. Conclusions

In this paper, we described the use of a hierarchical multi-resolution tracking framework that uses linear predictors. This allowed us to independently track in real-time a set of facial features with no constraints applied. Accurate and robust tracking is made possible by firstly introducing a novel biased-LP. Additionally, the visual tracking context for a facial feature is automatically identified using a novel probabilistic selection method. Experimental results based on convergence tests quantitatively show that the proposed method is more robust than existing AAMs. Crucially, this was achieved with a minimal training set of 13 to 15 images. We also show that the method tracks in the presence of occlusions due to spectacles, simultaneous pose and expression changes, as well as in conditions of low framerate and image quality.

Acknowledgements

The investigation reported here is supported by the EPSRC project LILiR.

References

[1] M. Barnard, E. Holden, and R. Owens. Lip tracking using pattern matching snakes. In *Proc. of the Fifth Asian Conference on Computer Vision*, January 2002.

[2] C. Bregler and S. Omohundro. Nonlinear manifold learning for visual speech recognition. In *Proc. of Fifth International Conference on Computer Vision*, pages 494–499, 1995.

[3] T. Cootes, G. Edwards, and C. Taylor. Active appearance models. *IEEE PAMI*, 23(6):681–685, 2001.

[4] P. Daubias and P. Deléglise. Statistical lip-appearance models trained automatically using audio information. *EURASIP J. Appl. Signal Process.*, 2002(1):1202–1212, 2002.

[5] J. Hoey. Tracking using flocks of features, with application to assisted handwashing. In *Proc. of British Machine Vision Conference*, pages 367–376, 2006.

[6] M. Kolsch and M. Turk. Fast 2d hand tracking with flocks of features and multi-cue integration. In *CVPRW '04: Proceedings of the 2004 Conference on Computer Vision and Pattern Recognition Workshop (CVPRW'04) Volume 10*, page 158, Washington, DC, USA, 2004. IEEE Computer Society.

[7] M. Lievin, P. Delmas, P. Coulon, F. Luthon, and V. Fristol. Automatic lip tracking: Bayesian segmentation and active contours in a cooperative scheme. In *Proc. of IEEE International Conference on Multimedia Computing and Systems*, volume 1, pages 691–696, July 1999.

[8] I. Matthews, T. Cootes, and J. Bangham. Extraction of visual features for lipreading. *IEEE PAMI*, 24(2):198–213, 2002.

[9] E. Ong and R. Bowden. Robust lip-tracking using rigid flocks of selected linear predictors. In *Proc. 8th IEEE Conf. on Automatic Face and Gesture Recognition*, 2008.

[10] I. Patras and E. Hancock. Regression tracking with data relevance determination. *IEEE Conference on Computer Vision and Pattern Recognition.*, pages 1–8, June 2007.

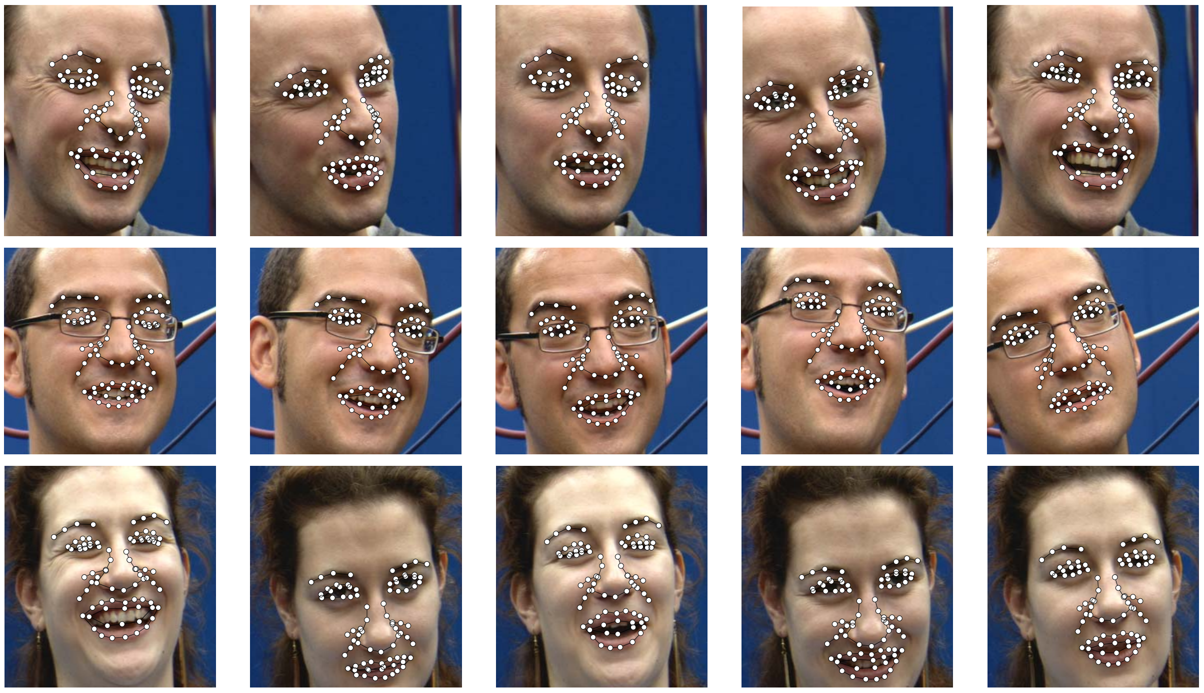

Figure 7. Results of tracking facial features using the SD camera sequences.

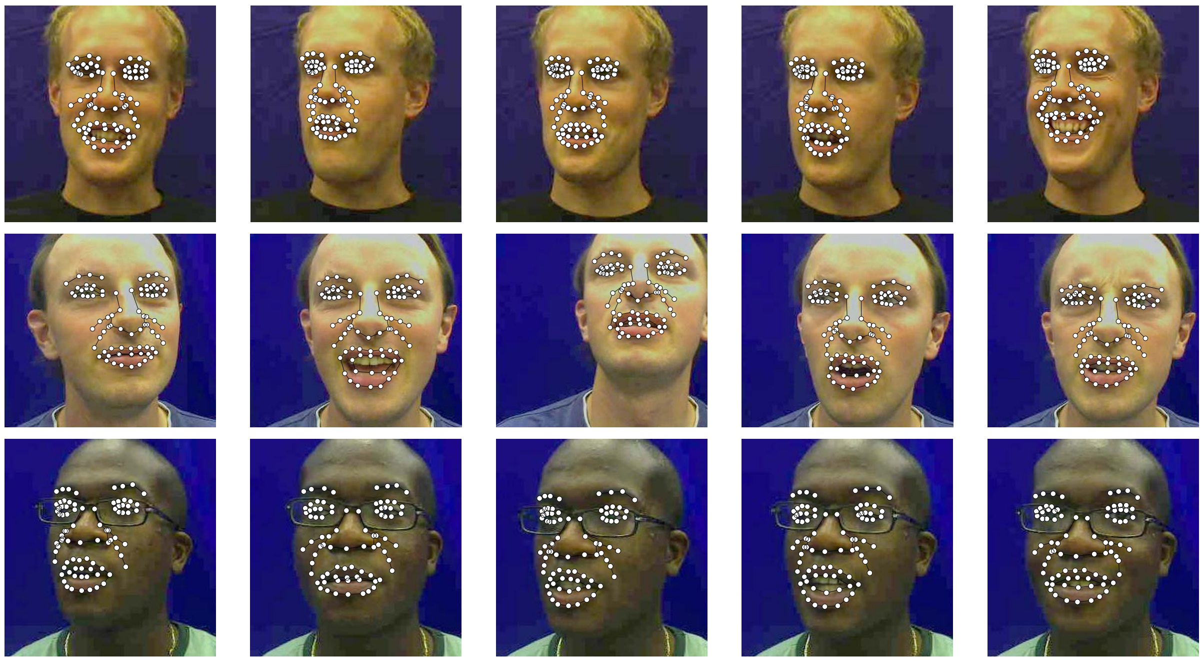

Figure 8. Results of tracking facial features using the webcam sequences.

[11] O. Williams, A. Blake, and R. Cipolla. Sparse bayesian learning for efficient visual tracking. *IEEE PAMI*, 27(8):1292–1304, 2005.

[12] Z. Wu, P. Aleksic, and A. Katsaggelos. Lip tracking for mpeg-4 facial animation. In *Proc. of the 4th IEEE Conf. on Multimodal Interfaces*. IEEE Computer Society, 2002.

[13] A. Yulle, P. Hallinan, and D. Cohen. Feature extraction from faces using deformable templates. *International Journal of Computer Vision*, 8(2):99–111, 1992.

[14] K. Zimmermann, J. Matas, and T. Svoboda. Tracking by an optimal sequence of linear predictors. *IEEE PAMI*, 31(4), 2009.