International Journal of Computer Vision Preprint

Linear Regression and Adaptive Appearance Models for Fast Simultaneous Modelling and Tracking

Liam Ellis · Nicholas Dowson · Jiri Matas · Richard Bowden

July 2010

Abstract This work proposes an approach to tracking by regression that uses no hard-coded models and no offline learning stage. The Linear Predictor (LP) tracker has been shown to be highly computationally efficient, resulting in fast tracking. Regression tracking techniques tend to require offline learning to learn suitable regression functions. This work removes the need for offline learning and therefore increases the applicability of the technique. The online-LP tracker can simply be seeded with an initial target location, akin to the ubiquitous Lucas-Kanade algorithm that tracks by registering an image template via minimisation.

A fundamental issue for all trackers is the representation of the target appearance and how this representation is able to adapt to changes in target appearance over time. The two proposed methods, LP-SMAT and LP-MED, demonstrate the ability to adapt to large appearance variations by incrementally building an appearance model that identifies modes or aspects of the target appearance and associates these aspects to the Linear Predictor trackers to which they are best suited. Experiments comparing and evaluating regression and registration techniques are presented along with performance evaluations favourably comparing the proposed tracker and appearance model learning methods to other state of the art simultaneous modelling and tracking approaches.

Keywords Regression Tracking · Online Appearance Modelling

This work is supported by DIPLECS, Dynamic Interactive Perception-action LEarning in Cognitive Systems, funded as contract number 215078 by the European Commission under FP7, and with support by Czech Science Foundation Project 102/07/1317.

L. Ellis
CVL, Linköping University, Linköping,Sweden
Tel.: +46 (0) 13 282 572
E-mail: liam@isy.liu.se

N. Dowson
AEHRC, Royal Brisbane and Women's Hospital, Brisbane, Australia
Tel: +61 (0) 7 3253 3641
E-mail: nicholas.dowson@csiro.au

J. Matas
CMP, Czech Technical University, Prague, Czech Republic,
Tel.:+42 (0) 2 2435 7212
E-mail: matas@cmp.felk.cvut.cz

R. Bowden
CVSSP, University of Surrey, Guildford,UK, Tel.: (+44)1483 300 800
E-mail: R.Bowden@Surrey.ac.uk

1 Introduction

This work is concerned with the development of fast visual feature tracking algorithms that utilise no prior knowledge of the target appearance. The approach presented here operates at high frame rates, tracks fast moving objects and is adaptable to variations in appearance brought about by occlusions or changes in pose and lighting. This is achieved by employing a novel, flexible and adaptive object representation comprised of sets of spatially localised linear displacement predictors associated to various modes of a multi modal template based appearance model learnt on-the-fly.

Conventional alignment based tracking approaches aim to estimate the position of the target in each frame by aligning an image template of the target with the new frame; the template (or input frame) is *warped* in order to obtain an optimal alignment. The warp parameters are obtained by optimising the registration

Table 1 Table of abbreviations used throughout text.

Abbreviations	Full meaning
LK	Lucas-Kanade: tracking by registration.
LP	Linear Predictor: tracking by linear regression.
SMAT	Simultaneous Modelling And Tracking: Adaptive multi-modal template based appearance model (Dowson and Bowden, 2006).
LK-SMAT	Tracking approach that combines LK displacement estimation with SMAT appearance modelling.
LP-SMAT	Tracking approach that combines LP displacement estimation with SMAT appearance modelling.
LP-MED	Tracking approach that combines LP displacement estimation with a medoid-shift based appearance model.

between the appearance model and a region of the input image according to some similarity function (*e.g.* L_2 norm, Normalised Correlation, Mutual Information). Optimisation is often carried out using gradient descent or Newton methods and hence assumes the presence of a locally convex similarity function with a minima at the true position. The *basin of convergence* of such methods is the locally convex region of the cost surface within which a gradient descent approach will converge. The size of the basin of convergence determines the *range* of the tracker *i.e.* the maximum magnitude of inter-frame displacements for which the approach will work. Trackers with small range require low inter-frame displacements to operate effectively and hence must either operate at high frame rates (with high computational cost) or only track slow moving objects. If the target moves a distance greater than the range between two consecutive frames then the method will fail. While multiscale approaches can be used to address this in registration approaches, regression based tracking allows the user to select the optimal range as a trade-off against accuracy and will be experimentally shown to have a greater range (not limited by the range of convexity or the presence of local minima in the cost surface) than registration methods and due to their simplicity are computationally efficient. The computational efficiency of the method is a result of learning a simple and general mapping directly from patterns of image intensity differences to desired displacements, and applying this mapping at each displacement prediction step, rather than performing an optimisation process for each prediction step.

Whilst prior models can be used to model target appearance, they place restrictions on the scope of applications for which the trackers can be easily used. Furthermore, visual tracking approaches that are able to adapt their representation of the target on-the-fly show increased robustness over approaches for which the representation is either specified (hard coded) or learned from a training set. Single template models, such as those employed in the Lucas-Kanade algorithm (Lucas and Kanade, 1981), aim to model the target appearance as one point on the appearance-space manifold. In order to increase robustness to appearance changes and minimise alignment drift, various template update strategies have been developed. These include naive update (Matthews et al, 2004) where the template is updated after every frame and strategic update (Matthews et al, 2004) where the the first template from the first frame is retained and used to correct location errors made by the updated template. If the size of the correction is too large, the strategic algorithm acts conservatively by not updating the template from the current frame. With template update methods, the template is intended to represent the current single point in the appearance-space manifold. Approaches that use some or all templates (Dowson and Bowden, 2006; Ellis et al, 2008), drawn from all frames, represent a larger part of this manifold. In this work, all stored templates are incrementally clustered to discover modes or *aspects* of the target appearance.

Tracking methods that adapt the representation of the target during tracking are prone to drift, as the appearance model may adapt to the background or occluding objects. The approaches proposed here address this problem by maintaining modes of an appearance model that correspond to past appearances. Whilst this approach reduces the impact of drift, as erroneous appearance samples do not contaminate all modes of the appearance model, the method does not address the drift verses adaptation trade-off directly. The work of Kalal et al (2010) explicitly addresses the trade-off between adaptation and drift.

The methods developed herein are designed to operate at high frame rates and as such need to be computationally efficient. The overarching design paradigm has been to use fast/simple methods: linear regression (for displacement prediction), random sampling (for learning displacement predictors and template extraction), incremental template clustering (for appearance modelling) and linear weighting (for associating displacement predictors with appearance modes). The use of simple regression methods is offset by an evaluation mechanism that allows both the weighting of the contribution of each displacement predictor and the continual disposal and replacement of poorly performing displacement predictors.

There are many different formulations of the tracking problem that lead to many and varied solutions:

tracking by detection (Viola and Jones, 2002), tracking using graph cut algorithms to iteratively segment the target (Bray et al, 2006) and condensation algorithms (Isard and Blake, 1998) to name but a few. Each approach is thought to have a certain scope of applications for which it will work best. It has been established that a significant class of tracking problems can be solved using the Linear Predictor and this paper aims to extend this class to problems requiring online feature tracking with appearance variation and real time operation. In particular, the approach presented here is, to the best of the authors knowledge, the first regression based tracking approach that continually evaluates and adapts the regression functions used for tracking on-the-fly. The experiments in section 7 go some way to delimiting the class of problems for which the proposed approach is suitable.

The rest of the paper is organised as follows: Section 2 contains a review of the relevant literature regarding the following three subjects; *tracking via registration*, *tracking via regression* and *online appearance model learning*. An overview of the proposed tracking methodology is then presented in section 3. Section 4 introduces two methods for learning models of the appearance of a target object during tracking and section 5 gives details of the registration and regression tracking approaches and some illustrative experimental results comparing the methods on real data are presented. In section 6 the complete tracking algorithms - that put together the regression techniques and the appearance modelling techniques - are presented. In section 7 a set of experiments are presented that characterise, compare and evaluate the proposed tracking approaches. Finally conclusions are discussed in section 8.

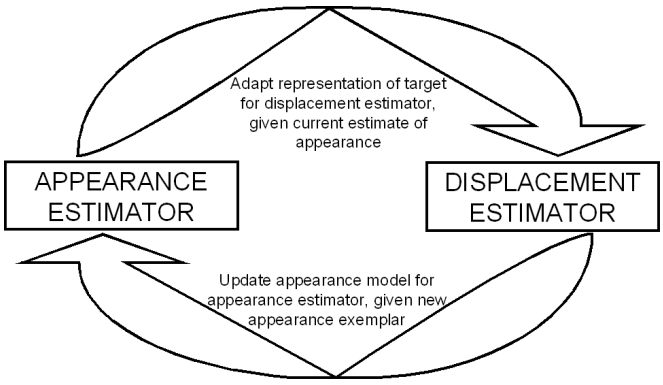


Fig. 1 *Simultaneous Modelling and Tracking methodology*: The displacement estimator, as well as generating the tracking output, provides a mechanism for supervision of the appearance model learning process *i.e.* it provides new examples of the target appearance that are added to the appearance model. In return, the appearance estimator provides information about the structure of the target appearance space that enables the tracker to cope with a high degree of variation in appearance.

2 Background

As this work is concerned with comparing two approaches (registration and regression) for predicting inter-frame displacement as well as techniques for combining these approaches with methods for learning appearance models online, this section contains a review of the relevant literature regarding the following three subjects; *tracking via registration*, *tracking via regression* and *online appearance model learning*.

2.1 Tracking via registration

Lucas and Kanade made one of the earliest practical attempts to efficiently align a template image to a reference image (Lucas and Kanade, 1981), minimising the Sum of Squared Difference similarity function. Efficiency was achieved by using a Newton-Raphson method in the space of warp parameters. In Newton-Raphson optimisation, iterative parameter updates to alignment parameters are obtained by multiplying the Jacobian by the inverse Hessian of the similarity function. Lucas and Kanade mainly considered translations, but later research considered more complex transformations and attempted to reformulate the similarity function allowing pre-computation of some terms. In particular, Hager and Belhumeur (1998) proposed inverting the roles of the reference and template at a strategic point in the derivation, and Shum and Szeliski (2000) constructed the warp as a composition of two nested warps. In a general treatise on Lucas-Kanade (LK) techniques, Baker and Matthews (2004) combined these methods to formulate the inverse-compositional method. Dowson and Bowden (2008) derived an inverse compositional formulation for aligning a template and a reference image using Mutual Information and Levenberg-Marquardt optimisation.

2.2 Tracking via regression

Cootes et al (1998) proposed a method for pre-learning a linear mapping between the image intensity difference vector and the error (or required correction) in AAM model parameters. Jurie and Dhome (2002) employed similar Linear Predictor (LP) functions to track rigid objects. The work of Matas et al (2006) again uses linear regression for displacement prediction, similar to the LP functions in (Jurie and Dhome, 2002) and (Cootes et al, 1998). They extend the approach by introducing

the Sequential Linear Predictor (SLP) (Zimmermann et al, 2009). Williams et al (2003) presented a sparse probabilistic tracker for real-time tracking that uses an RVM to classify motion directly from a vectorised image patch. The RVM extends the method of forming a regression between image intensity difference vectors and the error/correction to non-linear regression. Mayol and Murray (2008) extend these methods to general regression for tracking planar and near planar objects.

A key issue for LP trackers is the selection of its reference point, *i.e.* its location in the image. In the work of Marchand et al (1999) predictors are placed at regions of high intensity gradient but Matas et al (2006) have shown that a low predictor error does not necessarily coincide with high image intensity gradients. In order to increase efficiency of the predictors, a subset of pixels from the template can be selected as support pixels used for prediction. Matas et al (2006) present a comparison of various methods for learning predictor support, including randomised sampling and normalised reprojection, and found that randomised sampling is efficient with minimal and controllable tradeoff in terms of accuracy while Ong and Bowden (2009) employ an iterative learning scheme to choose optimal support regions for prediction.

This work avoids the need for costly reference point and support selection strategies by evaluating the performance of a predictor online and allowing poor performers to be replaced as opposed to minimising a learning error offline. Each of the displacement prediction trackers detailed in (Matas et al, 2006; Marchand et al, 1999; Williams et al, 2003; Ong and Bowden, 2009) require either an offline learning stage or the construction of a hard coded model or both. As shall be shown, this work does not require either hard coded models or offline learning. The approach in Mayol and Murray (2008), using generalised regression, can be trained at start up in a reported 0.5sec. However, once trained the method employs no online learning to adapt the regression functions.

Here the term 'online' implies that the learning is carried out on-the-fly, from a single frame drawn from the sequence during tracking. While the prediction function learning methods employed here are not incremental, they are less computationally expensive than other learning methods, and so can be employed at frame rate during tracking. The use of inexpensive learning methods results in potentially inaccurate prediction functions which necessitates the inclusion of mechanisms to evaluate, weight, remove and relearn the functions. Novel mechanisms for achieving this evaluation during tracking form an essential component of the proposed methodology.

2.3 Online appearance model learning

Tracking approaches typically employ appearance models in order to optimise warp parameters (*e.g.* translation or affine) according to some criterion function. Linear predictor trackers typically rely upon hard coded models of object geometry (Matas et al, 2006; Marchand et al, 1999). This requires significant effort in hand crafting the models and like simple template models (Lucas and Kanade, 1981; Baker and Matthews, 2004; Matthews et al, 2004), are susceptible to drift and failure if the target appearance changes sufficiently. Systems that use a priori data to build the model (Cootes et al, 1998) or train the tracker offline (Williams et al, 2003; Ong and Bowden, 2009) can be more robust to appearance changes but still suffer when confronted with appearance changes not represented in the training data. Incremental appearance models built online such as the WSL tracker of Jepson et al (2001) have shown increased robustness by adapting the model to variations encountered during tracking, but the overhead of maintaining and updating the model can prevent real-time operation. Ross et al (2008) have proposed an adaptive appearance model that incrementally learns a low dimensional appearance subspace representation, that operates at near frame rate (7.5Hz) and requires no offline training.

A number of methods have been proposed for online learning of discriminative feature trackers (Avidan, 2007; Collins et al, 2005; Grabner et al, 2006). The discriminative tracker of Grabner et al (2006) that uses an online boosting algorithm to learn a discriminative appearance model on-the-fly, achieves real-time tracking. Another entirely online approach that achieves real-time tracking is Dowson & Bowden's SMAT algorithm. Dowson and Bowden (2006) make a preliminary presentation of the Simultaneous Modelling And Tracking algorithm, SMAT, and show the benefits of online learning of a multiple component appearance model when employing alignment-based tracking.

3 System Overview

This section presents an overview of the proposed tracking architecture in general terms. In the following sections specific methods for each of the architectures components are introduced and evaluated.

At the most general level, the proposed tracking approach can be described by the following process:

1. **Estimate** the current target appearance using an appearance model

2. **Adapt** the displacement estimation mechanism to suit current estimate of appearance
3. **Estimate** inter-frame displacement of the target
4. **Adapt** the appearance model given new appearance data
5. **Repeat** steps 1-4

These stages are achieved by the interaction of two components, namely the **displacement estimator** and the **appearance estimator**. The displacement estimator, as well as generating the tracking output, provides a mechanism for supervision of the appearance model learning process *i.e.* it provides new examples of the target appearance that are added to the appearance model. In return, the appearance estimator provides information about the structure of the target appearance space that enables the tracker to cope with a high degree of variation in appearance. This basic methodology is represented in figure 1.

The target appearance samples - templates drawn from the image at the targets estimated location - will change during tracking. This is caused by all appearance variations that are not modeled by the pose parameters *e.g.* rotation (if translation transformations only are considered), lighting change, occlusion or changes of expression (when tracking faces) as well as frame-to-frame inaccuracies in displacement estimation. In the proposed appearance modelling approach, all stored templates are incrementally clustered to discover modes or *aspects* of the target appearance. Identifying the current aspect of the target appearance is the role of the appearance model, as shown in figure 2. By identifying aspects of the target, it becomes possible to adapt the displacement estimation mechanisms to suit the current appearance.

The proposed tracking framework associates these aspects to banks of displacement estimators - trackers - via an *association matrix*, see figure 2. The values in the association matrix reflect the suitability of each tracker to each aspect of the target. This provides a flexible way of controlling the influence of each tracker to the overall pose estimation.

Within this architecture there are many possible approaches to implementing the appearance model, association strategy, displacement estimation and final pose estimation processes. In section 4 two methods for on-the-fly appearance modelling are introduced. Two displacement estimation methods - template registration and linear regression - are investigated in section 5. In section 6 various configurations of the complete tracking framework are detailed.

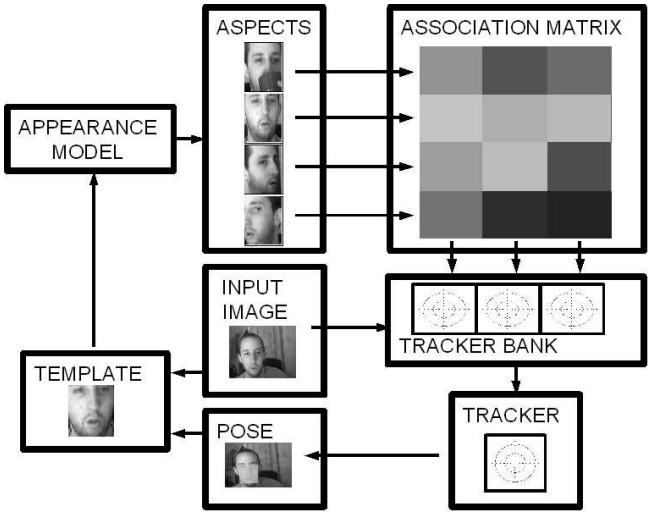


Fig. 2 *Generic system architecture*: The appearance model stores all target templates and identifies aspects of the target. Aspects are associated to feature trackers by an association matrix. Each feature tracker contributes to the overall pose estimation, the level of contribution is determined by the strength of association to the current aspect *i.e.* the association matrix value.

4 Adaptive Appearance Models

Aside from the intrinsic requirement of a representation of the target appearance for all tracking methods, appearance models can additionally help cope with appearance changes not parameterised by the pose parameters. Provided a perfect geometric model of the target *and* environment was available, it would be possible to parameterise every possible change to the target appearance. Such a model would have to include parameterisations of not only all degrees of freedom (DOF) of the target object but also other objects in the environment that may occlude the target along with environmental effects such as changes in lighting. This is simply not feasible in any real scenario. In addition, the estimation of the huge number of parameters required by such a model would be intractable. Tracking approaches, therefore, tend to model only a subset of pose parameters, commonly translation (2 DOF) or affine (6 DOF). Any changes not represented by the selected pose parameters will often cause tracking failure. An appearance model can provide a means of compensating for this partial parameterisation.

4.1 Aspect learning for tracking

Both regression and registration based trackers, that are intended to track a 3D object such as, for example, a cube, are initialised by identifying the region in the first frame that contains the cube. If the cube then

starts to rotate, perhaps exposing a new face of the cube and hence presenting a new *aspect* of the target, the initial target representation may no longer be adequate. It would therefore be advantageous to identify that a new aspect of the target had been presented and to adapt the target representation used for tracking accordingly. Eventually the cube may rotate back to its original orientation and thus present the initial aspect of the target again. In this case it would be advantageous to recall the representation associated to that aspect. This is the function of the appearance models developed here: to identify different aspects of the target - clusters of appearance samples - such that the target representation used in estimating inter-frame displacement can be partitioned and associated with the aspects for which they perform well.

The term 'aspect' is used to describe some mode or cluster of the appearance manifold. As discussed above, the appearance manifold may include regions associated with all appearance variations not modeled by the pose parameters *e.g.* rotation, lighting change, occlusion or changes of object appearance itself.

If a single template appearance of an object is considered as one point on the appearance-space manifold (as in the Lucas-Kanade method), the manifold can be represented by storing the set T of all templates, $T = \{\mathbf{G}^0 ... \mathbf{G}^t\}$ drawn from all frames $\{\mathbf{F}^0 ... \mathbf{F}^t\}$. In order to identify aspects of the target, the set of templates, T, should be clustered or partitioned, $T = \{T^0 ... T^M\}$ where $T^m \subset T$.

For a subset of templates, $T^m \subset T$, to represent a real aspect of the target appearance, the templates that make up an aspect should be similar to one another and different to the templates in all other aspects. Similarity is determined by a distance metric. The L_2 norm distance is used in the below methods due to its computational efficiency but others, such as Mutual Information or Normalised Correlation could also be used. Both the clustering methods detailed below compute and maintain a matrix of L_2 norm distances between templates and use this to determine each templates aspect membership *i.e.* to which aspect that template belongs.

4.2 SMAT: Greedy template clustering

In order to identify different aspects of the target, modes or clusters of the appearance manifold must be discovered. The method presented here partitions the appearance manifold, assigning templates to partitions with a greedy incremental algorithm.

Each of the M aspects, $T^m \subset T$, $m = 1...M$ of the appearance manifold are represented by: a group of templates, the median template μ^m, a threshold τ^m, and a weighting w^m. Use of the median rather than the mean avoids pixel blurring caused by the averaging of multiple intensity values of templates that are not perfectly aligned. Weight w^m represents the estimated a priori likelihood that the m^{th} partition best resembles the current appearance of the target. During tracking, a template is drawn from the new frame at the location determined by the displacement estimator. To identify the best matching partition to the new template, a greedy search is performed starting with the partition with the highest weight and terminating when a partition, T^{m^*}, is found whose L_2 norm distance to the image patch is less than the threshold τ. The input image patch is then added to partition T^{m^*} and the median, μ^{m^*}, threshold, τ^{m^*}, and weights, $w^m, m = 1...M$, are updated. See Eq. 1 for the component weight update strategy. If no match is made, a new component is created with the new template and the template from the previous frame. The learning rate, α, sets the rate at which component rankings change and is set to α=0.2 for all experiments. This value was found through experimentation.

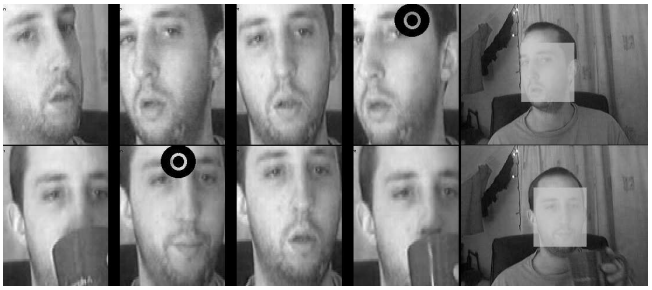


Fig. 3 *Appearance model medians for the head tracking sequence*: Two examples of the median templates of the four partitions of the appearance space are shown, ordered with decreasing weight from left to right. It is clear that the modes identify aspects of the target such as side/front/occluded views. The matched component for the current frame is marked with the bullseye.

$$w^m = \begin{cases} \frac{w^m + \alpha}{1+\alpha} & \text{if } m = m^*; \\ \frac{w^m}{1+\alpha} & \text{if } m \neq m^*. \end{cases} \quad (1)$$

To facilitate the efficient update of an appearance model component, a matrix $\mathbf{Q}^m$ maintains the L_2 norm distances between each pair of templates in the m^{th} component. Adding a new template to the component then requires only the computation of a single row of $\mathbf{Q}^m$ *i.e.* the distances between the new template and all other templates. The median template index, j^*, is calculated using Eq. 2 and the component threshold τ^{m^*} is computed using Eq. 3 which assumes an approximately Gaussian distribution of distances and sets the threshold to three standard deviations of the distribution.

$$j^* = \underset{j}{\operatorname{argmin}} \sum_{i=0}^{n} Q_{ij}^m, j = 1....n \quad (2)$$

$$\tau^{j^*} = 3\sqrt{\frac{1}{n}\sum_{i=0}^{n}(Q_{ij^*}^m)^2} \quad (3)$$

The dimensions of $\mathbf{Q}^m$ depend on the number, n, of templates in the model but can be limited to bound memory requirements and computational complexity. In practice, new templates replace the worst template from the component. It is also possible to limit the number of components, M. When creating a new component the new component replaces the worst existing component identified by the lowest weight $m_{worst} = \underset{m}{\operatorname{argmin}}\, w^m, \{m = 1...M\}$.

For all the experiments presented in section 7.4 a maximum of n=60 templates are maintained in each of a maximum of M=4 components of the model. This is found to be sufficient to model a stable distribution whilst preventing computational costs becoming too high for real-time tracking. Figure 3 illustrates the SMAT model being used to identify aspects of a head during a head tracking sequence. It can be seen that the modes identify aspects of the target such as side, front or occluded views.

4.3 Medoidshift template clustering

Fig. 4 The distance matrix pre and post clustering is shown with three subsets of exemplars **A**, **B** and **C**. Sets **A** and **C** are temporally separated but have the same appearance. Templates from each subset are also shown.

The second appearance model presented is again constructed online by incrementally clustering image patches to identify various modes of the target appearance manifold. Here, the clustering is performed by the medoidshift algorithm introduced by Sheikh et al (2007). Medoidshift is a nonparametric clustering approach that performs mode-seeking by computing shifts toward areas of greater data density using local weighted medoids. As Sheikh et al (2007) show, the procedure can be performed incrementally, meaning the clustering can be updated at the inclusion of new data samples and the removal of some existing data samples.

During tracking the appearance templates are collected into vectors $\{\mathbf{G}^0...\mathbf{G}^t\}$ and, as for the greedy clustering approach, a distance matrix, $\mathbf{Q}$ is populated with the L_2 norm distances. Where the SMAT model maintains a $\mathbf{Q}$ matrix for each model component, this model maintains one $\mathbf{Q}$ matrix recording the distance between each stored frame. The medoidshift algorithm uses $\mathbf{Q}$ to obtain a clustering[1]. The clustering is incrementally updated given a new $\mathbf{G}$ vector and hence (by computing L_2 norm values) a new row/column of $\mathbf{Q}$. In order to constrain the memory requirements and computational complexity of maintaining the appearance model, the number of templates retained, and hence the number of data points clustered, is limited. Once the limit has been reached the oldest template is removed and replaced with the new template. The cluster update must accommodate both the introduction and removal of data points. The incremental update is achieved in a computationally efficient manner exactly as described in (Sheikh et al, 2007).

The effect of this clustering, illustrated in figure 4, shows the distance matrix at frame 275 of a head tracking sequence before and after matrix indices are sorted according to the cluster label. As can be seen, two temporally separated subsets, A and C, of templates are assigned to the same cluster, $A \cup C \subset T$, identifying the *front view* aspect whilst a third subset, $B \subset T$, is partitioned and identifies a *side view* aspect of the face. It is obvious that a displacement estimator that represents the target appearance of the hidden side of the face will be less reliable while this side view aspect is presented.

4.4 Appearance model discussion

While the greedy approach provides a computationally efficient method of partitioning the templates $T = \{\mathbf{G}^0...\mathbf{G}^t\}$ into aspects, $T = \{T^0...T^M\}$ where $T^m \subset T$, the algorithm lacks some flexibility. Rather than the number of aspects being a predefined value, M should ideally be data dependent and reflect (rather than determine) the number of modes present in the data's distribution. Also once a template is assigned to a certain partition it will never become part of another partition. This rigidity in terms of template-to-cluster assignment and fixed number of modes is likely to cause problems as the target appearance manifold evolves during tracking.

The data driven, mode seeking medoidshift incremental clustering algorithm offers greater flexibility to

[1] As no meaningful partitioning is possible with small sample sets, the clustering procedure is not carried out until frame 11 of tracking

the appearance modelling process. The number of aspects, M, are not predefined and, as the appearance manifold grows and changes over time, so too can the aspect membership of each template.

Whilst the flexibility of the medoidshift approach allows a representation that is more reflective of the real underlying appearance distribution, the resulting representation of the aspects are less straightforward to interpret than the SMAT model. As the SMAT model has a fixed number of aspects, it is straightforward to construct an association matrix that associates a set of displacement predictors to each of the models aspects. With the medoidshift approach however, the varying number of aspects discovered and the adaptive cluster membership of templates necessitates a less straightforward association mechanism. Section 6 gives details of how both the appearance models are used within the tracking framework.

A significant factor in the computational overhead of these appearance models is the maintenance of the distance matrix, $\mathbf{Q}$. As stated, this can be controlled by limiting the number of templates stored by the model. Another way to control the computational cost is to reduce the dimensionality of the distance function *i.e.* to subsample the image templates prior to computation of L_2 norm distances.

5 Regression vs. Registration

This section details and compares the registration and regression approaches to predicting inter-frame displacement of a target object for tracking. First, details of the registration and regression tracking methods are given. The Linear Predictor (LP) regression tracker is introduced and the method used for learning the LP regression function is detailed followed by a description of methods of combining the outputs of multiple LPs - LP flocks. Finally some experimental results are presented that compare regression and registration techniques on an example of inter-frame displacement prediction.

The tracking problem is defined as the task of estimating the change of pose or warp parameter, $\delta\mathbf{x}$, such that:

$$I_R(W(\mathbf{x}, \delta\mathbf{x})) \approx I_T \tag{4}$$

Where I_R is the new input image, W is a warping function (*e.g.* translation, affine) and I_T is a template representing the appearance of the target.

For the LK or registration based method this is treated as a minimisation problem such that we wish to find $\delta\mathbf{x}$ that minimises the dissimilarity between I_R and I_T.

$$\delta\mathbf{x} = \underset{\delta\mathbf{x}}{\operatorname{argmin}} \, ||I_R(W(\mathbf{x}, \delta\mathbf{x})) - I_T|| \tag{5}$$

For the regression or Linear Prediction (LP) method, the prediction directly estimates $\delta\mathbf{x}$.

$$\delta\mathbf{x} = \mathbf{P}(I_R(W(\mathbf{x}, \mathbf{dx})) - I_T) \tag{6}$$

Every tracking approach has some representation of the target; tracking output is a function of both this representation and new image data. For registration methods the representation is a template of pixel intensities, I_T, drawn from the input image at the location of the target. Tracking is then the process of aligning template, I_T, with the new input reference image, I_R *i.e.* finding the warp, W, with parameters $\delta\mathbf{x}$ that minimises (maximises) some distance (similarity) function between I_T and I_R.

For the linear regression method presented in section 5.2 the target representation is a vector of image intensities. Additionally, the regression function, $\mathbf{P}$, encodes information about the target appearance. Tracking is then the process of multiplying $\mathbf{P}$ with the difference between target representation vector and an intensity vector sampled from the input image at the current position.

Looking at equations 5 and 6 it is apparent that both approaches involve some operation on the difference between the target representation and the input image information. Whilst the registration method explicitly minimises the cost surface to obtain an optimal alignment, the regression method directly maps from image intensity difference patterns to required displacements. In fact, as detailed in the following section, the iterative optimisation methods used in the registration approaches involve, at each iteration, a linear operation on the intensity difference. The difference between the two methods is that the parameters of the linear function used in the iterative optimisation methods are based on cost surface gradient information, whereas for the regression methods, the parameters are learnt from examples of displacement and intensity difference patterns.

5.1 Tracking by registration

The registration process aims to locate the region in I_R (reference image) that most resembles I_T (template image) by minimizing a distance function, f, which measures the similarity of the two regions. The position of I_T relative to I_R is specified by a warp function W with parameters $\delta\mathbf{x}$.

$$\delta\mathbf{x} = \underset{\delta\mathbf{x}}{\operatorname{argmin}}\ f[I_R(W(\mathbf{x}, \delta\mathbf{x})), I_T(\mathbf{x})] \tag{7}$$

Distance function, f, can be any similarity measure, *e.g.*, L_2 norm or MI. For comparisons of the relative merits of different similarity measures see (Dowson and Bowden, 2008). The position of greatest similarity is found using an optimisation method. LK methods use a group of optimization methods, the so-called Newton-type methods, *i.e.* methods which assume locally parabolic shape and proceed with an update as follows:

$$\delta\mathbf{x}^{(k+1)} \leftarrow \delta\mathbf{x}^{(k)} - H^{-1}(\delta\mathbf{x}^{(k)})G(\delta\mathbf{x}^{(k)}) \tag{8}$$

Where H, $\frac{\partial^2 f}{\partial \delta\mathbf{x}^2}$, is the Hessian of f, and G, $\frac{\partial f}{\partial \delta\mathbf{x}}$, is the Jacobian, while k indexes the iteration number. However, minima in tracking and registration problems are frequent which results in erroneous alignment of the template with the target. Multiple initializations can improve performance but at an obvious computational cost.

Generally, LK type methods apply Quasi-Newton optimisation, *i.e.* an approximation to the Hessian, $\tilde{H}$, is used. In general, Newton and Quasi-Newton only perform well when near to the minimum. Steepest Descent methods, which ignore local curvature and instead multiply G by a scalar *step-size* value λ, perform better when further from the minimum. The Levenberg-Marquardt (Marquardt, 1963) method combines these two methods. In this work a formulation similar to that presented in (Dowson and Bowden, 2008) (using Levenberg-Marquardt and L_2 norm) of this registration based tracking is used in comparisons with regression based techniques. The C++ (or Matlab) warthog library is used as an efficient implementation[2].

5.2 Tracking by regression

Feature tracking by regression is achieved by predicting inter-frame displacement of the target. The displacement predictors explored here use linear models to predict. These predictors compute motion at a reference point from a set of pixels sub-sampled from its neighbourhood called the support set $\mathbf{S} = \{\mathbf{s}_1, ..., \mathbf{s}_k\}$. The intensities observed at the support set $\mathbf{S}$ are collected in an observation vector $\mathbf{l}(\mathbf{S})$. The $\mathbf{l}_0(\mathbf{S})$ vector contains the intensities observed in the initial training image. Here the motion is a 2D translation $\mathbf{t}$, we use

[2] Link to code found at www.cvl.isy.liu.se/research/adaptive-regression-tracking

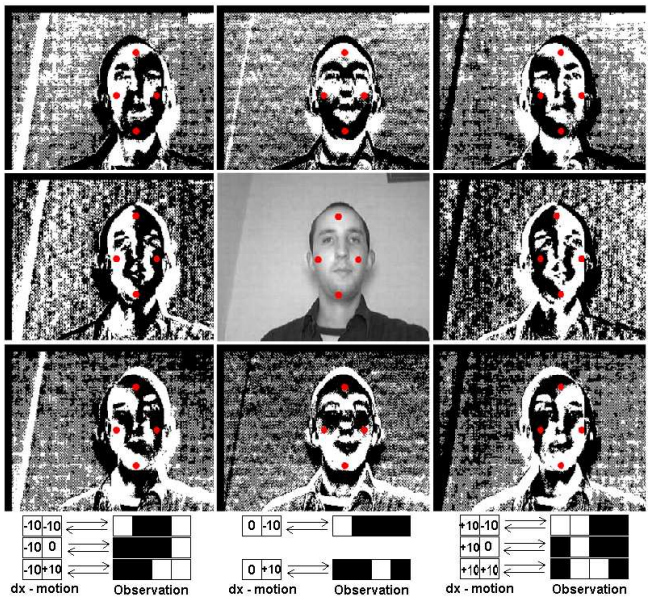


Fig. 5 Intensity difference images for eight translations. Four support pixel locations illustrate the predictive potential of the difference image. The input image is in the center. All images to the left/right of the input have been translated left/right by 10 pixels. Those images above/below the input have been translated by 10 pixels up/down. Under the images, the motion and support vectors are illustrated.

$(\mathbf{S} \circ \mathbf{t}) = \{(\mathbf{s}_1 + \mathbf{t}), ..., (\mathbf{s}_k + \mathbf{t})\}$ to denote the support set transformed by $\mathbf{t}$. Translation is sufficient as the multi-modal appearance models developed in section 4 cope with affine deformations of the image templates, also shown in (Dowson and Bowden, 2006).

Predictions are computed according to the expression in Eq. (9) where $\mathbf{P}$ is a $(2 \times k)$ matrix that forms a linear mapping $\Re^k \rightarrow \Re^2$ from image intensity differences, $\mathbf{d} = \mathbf{l}_0(\mathbf{S}) - \mathbf{l}(\mathbf{S} \circ \mathbf{x})$, to changes in warp parameters, $\delta\mathbf{x}$. The state vector, $\mathbf{x}$, is the 2D position of the predictor after prediction in the preceding frame.

$$\delta\mathbf{x} = \mathbf{P}\mathbf{d} = \mathbf{P}(\mathbf{l}_0(\mathbf{S}) - \mathbf{l}(\mathbf{S} \circ \mathbf{x})) \tag{9}$$

This efficient prediction only requires k subtractions and a single matrix multiplication, the cost of which is proportional to k.

5.3 Predictor learning

In order to learn $\mathbf{P}$, the linear regressor or projection matrix, N training examples of $\{\delta\mathbf{x}_i, \mathbf{d}_i\}$ pairs, $(i \in [1, N])$ are required. These are obtained from a single training image by applying synthetic warps and subtracting the deformed image from the original. For efficiency, the warp and difference computation is only

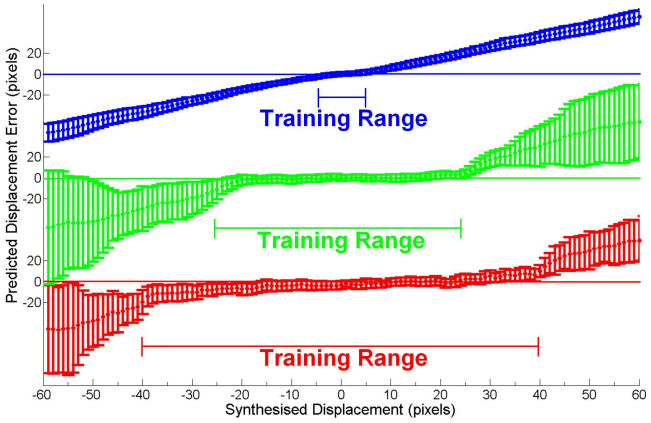


Fig. 6 The predicted displacement error (vertical axis) versus the true displacement (horizontal axis) of three LPs is shown. The response shown in red (or dark grey in black and white) at the bottom is of a predictor trained on displacements in the range -40 to 40 pixels. The response shown in green (light grey) in the middle is of a predictor trained on displacements in the range -25 to 25 pixels and the response shown in blue (black) at the top is of a predictor trained on displacements in the range -5 to 5 pixels. It can be seen that, within the range of displacements used for training, each of the LPs achieve relatively low errors. It can also be seen, from the error bars, that whilst increasing the range of displacements used for training extends the operational range of the LP, it does so at the cost of stability.

performed at the support pixel locations but, for illustration, the result of performing this operation on the entire image is shown in figure 5 for eight different translation warps. Also marked on the figure are four possible locations for support pixels and the unique observation patterns they produce.

In this approach, support pixels are randomly selected from within a range, r_{sp}, of the predictors reference point. This is in contrast to other LP learning strategies (Zimmermann, 2008; Ong and Bowden, 2009) where the objective is to select an optimal support set. The next step in learning the linear mapping $\mathbf{P}$ is to collect the training data, $\{\delta\mathbf{x}_i, \mathbf{d}_i\}$ into matrices $\mathbf{X}$, $(2 \times N)$, and $\mathbf{D}$ $(k \times N)$ where N is the number of training examples. The Least SQuares (LSQ) solution, denoted P, is then:

$$\mathbf{P} = \mathbf{X}\mathbf{D}^{+} = \mathbf{X}\mathbf{D}^{T}(\mathbf{D}\mathbf{D}^{T})^{-1} \tag{10}$$

Where $\mathbf{D}^{+}$ is the pseudo inverse of $\mathbf{D}$. Clearly there are more sophisticated learning methods, both in the selection of support pixels and in the method used to solve the regression problem. However, the methods selected provide a computationally efficient solution. As shall be shown here and in section 4, the use of LPs with low computational cost combined with methods to rate the performance (and hence weight the contribution) of each LP allows the replacement of poorly performing LPs during tracking. This essentially spreads the cost of learning appropriate mappings over a period of time and allows incremental learning as opposed to batch (offline) learning.

The LPs have a number of tunable parameters, these are listed along with the values used in table 2. The parameter, r_{sp}, defines the range from the reference point within which support pixels are selected. Parameter r_{tr} defines the range of synthetic displacements used for training the predictor. Figure 6 illustrates the displacement prediction errors of LPs with $r_{tr} = 10$, $r_{tr} = 50$ and $r_{tr} = 80$. The predictor complexity, k, specifies the number of support pixels used and hence the dimension of $\mathbf{P}$. The number of synthetic translations used in training is denoted N. In section 7.2, experimental results are presented to illustrate the effect each of these parameters has on tracker performance. It is sufficient to say, increasing r_{tr} increases the maximum inter frame displacement at the expense of alignment accuracy; k models the trade off between speed of prediction and accuracy/stability. N does not affect prediction speeds but instead parameterises a trade off between predictor learning speeds and accuracy/stability.

5.4 The linear predictor flock

The displacement predictions made by LPs have limited accuracy; this is especially the case where no attempt is made to optimise support pixel selection. A simple approach to handling the noise introduced by this inaccuracy is to take the mean prediction from a collection of LPs as in equation 11.

$$\delta\bar{\mathbf{x}} = \frac{\sum_{l=1}^{L} \delta\mathbf{x}^{l}}{L} \tag{11}$$

The state vector, x for each of the collection of L LPs is then updated with this mean prediction, as in equation 12, causing the LPs to *flock* together.

$$\mathbf{x}_t^l = \mathbf{x}_{t-1}^l + \delta\bar{\mathbf{x}}, l = 1...L \tag{12}$$

The increase in prediction accuracy, as shown by the experiments in section 7.4, is due to the noise averaging characteristics of the mean. Similar results are/would be obtained using the median but this would complicate the weighting of LP contribution to flock output as described below. Another approach is to use the RANSAC algorithm to select the subset of LPs who's prediction gains most consensus within the flock. Although the outlier rejection of RANSAC may be better than the mean value, RANSAC has a higher computational cost

and again is less well suited to weighting LP contributions to flock output.

Within an LP flock, it is desirable to down weight poor predictions or even remove/replace poorly performing LPs. This is especially the case when using the simple learning strategies detailed above. To weight the contribution a single prediction makes to the overall flock output, some way of assessing the reliability of the prediction is required - a prediction error. As no ground truth displacement is available whilst tracking, this error function could rely on observation differences at the support pixels, the assumption being that when a predictor performs well, the observations at the support pixels - after the trackers state vector, x, has been updated - should be similar to those observed in the initial frame. Alternatively, we can consider flock output to be 'truth' and evaluate predictions based on flock agreement, *i.e.* the error is the difference between 'true' flock output and the prediction being evaluated. If an LP 'strays from the flock' it can be relied on less. This approach benefits from its computational simplicity as it requires only difference computations in the low dimensional pose space, $\|\delta\bar{\mathbf{x}} - \mathbf{x}_t^l\|$ (t is the current frame) as opposed to in the higher dimensional observation space $\|\mathbf{l}_0^l - \mathbf{l}_t^l\|$. The observation difference error also requires additional computation for image bounds checking.

There is considerable scope for different LP flock contribution weighting strategies using either of the above prediction errors. A simple and cost effective approach is linear weighting (see equation 14) with normalised errors (see equation 13). The weighting can be based on the errors computed in the current frame, the previous frame or, as investigated in section 4, the history of the LP's performance. In section 6, the weightings are computed in such a way as to control the contribution of predictors dependent on its usefulness given the current appearance of the target. Equation 13 shows how a weight is computed and equation 14 illustrates the linearly weighted LP flock.

$$w^l = 1 - \frac{\|\delta\bar{\mathbf{x}} - \mathbf{x}_t^l\|}{\max \|\delta\bar{\mathbf{x}} - \mathbf{x}_t^l\|, l = 1...L} \tag{13}$$

$$\delta\bar{\mathbf{x}} = \frac{\sum_{l=1}^{L} (w^l \cdot \delta\mathbf{x}^l)}{\sum_{l=1}^{L} w^l} \tag{14}$$

The experiments in section 7.3 show how this weighting strategy improves the accuracy of the LP flock.

The ability to control the level of each LP's contribution to the overall tracking output enables a high level of adaptability and flexibility to the feature tracker - LPs can be associated to various aspects of the target feature as in section 6. Furthermore, evaluating each LP's performance provides the possibility to discard LPs that consistently perform poorly. The process of evaluating, discarding, re-learning and weighting performs a similar optimisation process to that performed in offline training approaches or registration processes such as the Lucas-Kanade tracker, but it does so incrementally whilst the tracker is operating.

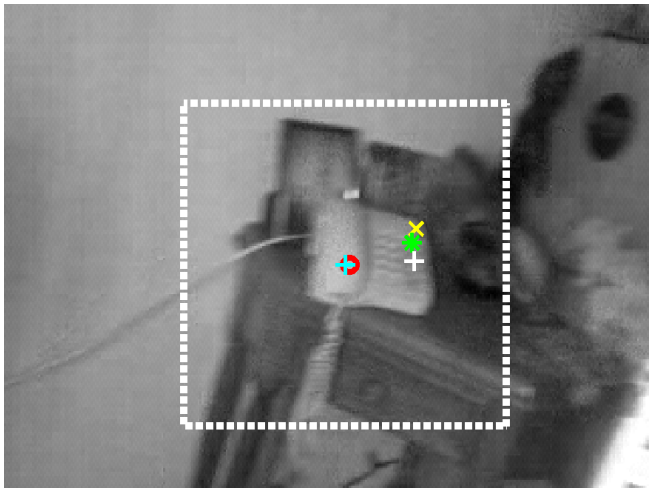
5.5 Inter-frame motion example

Each of the three trackers under investigation (Lucas-Kanade, LP and LP flock) was applied to an image sequence, captured from a moving web camera, containing considerable motion blur and large inter-frame displacements caused by vigorous shaking of the camera. Figures 7(a) and 7(b) show frames 374 and 375 respectively.

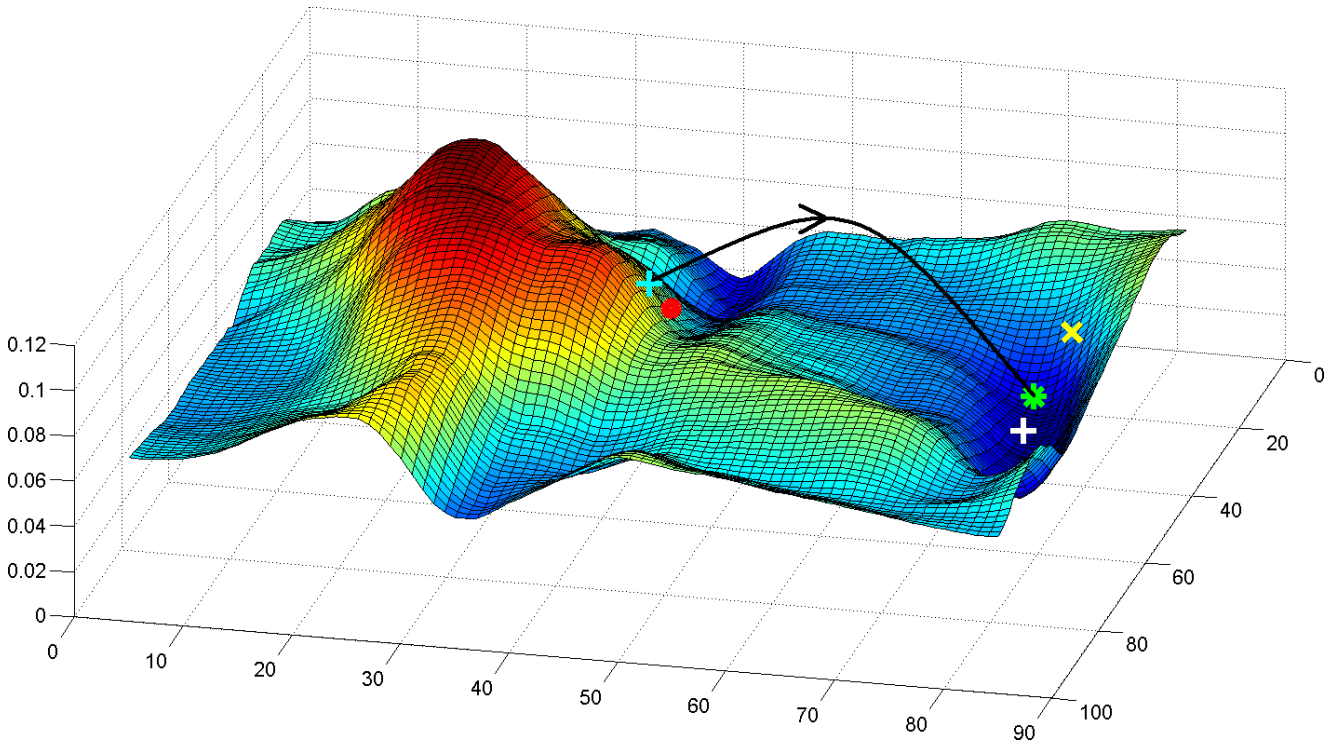
On figure 7(a) the reference point being tracked is indicated by the cross. On figure 7(b), which shows frame 375 as suffering considerable motion blur, the same co-ordinate is marked in light blue (grey in black and white) cross. Also marked on figure 7(b) is the position each of the trackers believes to be the target. The Lucas-Kanade tracker (red circle) has moved a short distance from the position in the previous frame and has failed to track the target. The single LP tracker (yellow X) has done better and the LP flock (green star) has done better still. The 'true' point (white cross) is obtained by taking a template of the target and finding the global minimum in the cost surface as shown in figure 7(c).

Figure 7(c) is informative as it illustrates the difference between the regression and registration processes, specifically highlighting the problems of using gradient descent or Newton methods that assume the presence of a locally convex similarity function with a minima at the true warp position. Although the global minimum of the similarity function, or cost surface, is at the true warp position, the Lucas-Kanade tracker is 'caught' in a local minimum. The inter-frame displacement was larger than the basin of convergence of the tracker *i.e.* it fell outside the area of convexity of the surface around the true point. On the other hand, both the regression techniques are able to 'leap' across the cost surface and track successfully despite motion blur and the large 37 pixel inter-frame displacement. This is because the regression approach learns how patterns of image intensity differences relate to displacements. In section 7 various trackers, including more advanced registration based approaches, are tested on the entire 1000 frame video sequence and, due mainly to severe

(a) Frame 374 from video with camera motion. Location of reference point of feature being tracked is marked with a cross. The template size (40 by 40 pixels) is also marked with a rectangle.



(b) Frame 375. Location of reference point in last frame marked with a light blue (grey in black and white) cross. LK point of convergence marked with a red (dark grey) circle, single LP with a yellow (white) X, LP flock with a green (black) star and the true position marked with a white cross. The search area used to produce the cost surface below is marked with a white rectangle.



(c) Cost surface of L_2 norm distance between template drawn from frame 374 and an 80 by 80 pixel region of frame 375 around reference position in frame 374. Light blue (grey in black and white) cross indicates position in frame 374, red (black) circle is the location the Lucas-Kanade algorithm converges to, yellow (grey) X is the single LP result, green (grey) star is the LP flock result and the white cross is the global minimum of the cost surface that corresponds to the true position in frame 375.

Fig. 7 *Inter-frame motion example*: The registration (circle), regression (X) and flock (star) displacement estimators are tested on a image sequence featuring vigorous camera shake. The regression methods are shown to accurately estimate the large (37 pixel) inter-frame displacement while the registration method fails due to a local minimum in the cost surface.

camera shake and hence large inter-frame displacement, only the regression methods are successful.

6 Tracking Framework

This section details three configurations of the tracking framework: LK-SMAT, LP-SMAT and LP-Medoidshift. The first method uses the appearance model to identify different aspects of the target appearance and to provide a template - the median template of the best matching model component - for use in the registration process. For the LP methods, the function of the appearance models developed is to identify different aspects of the target such that the set of LPs can be partitioned and associated with the aspects for which they perform well.

Details of the mechanisms used to associate flocks of LPs with appearance modes identified by each of the appearance models are presented. Due to differences in the clustering approaches used - greedy and medoidshift clustering - different strategies for this partitioning and association are required. Specifically, with the medoidshift approach, there is not a fixed number of modes and an appearance template may change the cluster to which it belongs, whereas with the SMAT approach, there is a fixed number of modes and a template is assigned to just one mode for the duration of tracking.

6.1 LK-SMAT: Registration based Simultaneous Modelling and Tracking

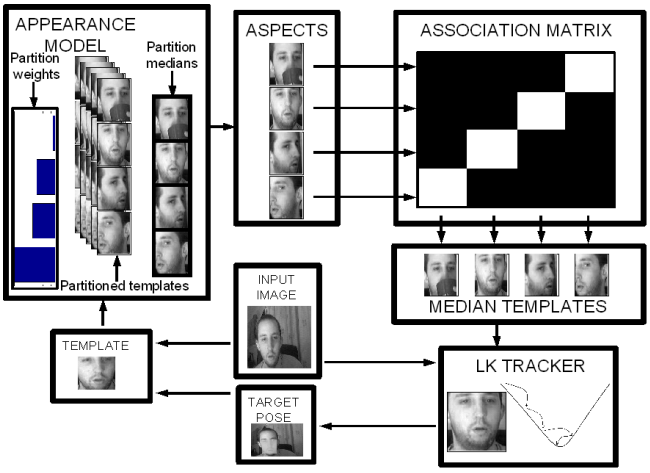


Fig. 8 *LK-SMAT system architecture*: The SMAT appearance model identifies aspects by partitioning templates using a greedy clustering algorithm. Identifying the current aspect selects the template for use in registration process. The association matrix in this formulation is simply an identity matrix.

Algorithm 1 LK-SMAT tracking procedure

$\mathbf{F}^0 \leftarrow$first image
Initialise target position $\bar{\mathbf{x}}^0$, height h and width w from user input
Extract first appearance template $\mathbf{G}^0$
Set initial component weight $w^m = 1$
while $\mathbf{F}^t \neq NULL$ **do**
  Register currently selected appearance template $\mathbf{G}^*$ with new frame $\mathbf{F}^t$ as in eq. 7
  Extract new appearance template $\mathbf{G}^t$ at estimated target location
  Assign new template to partition m^* using greedy search algorithm
  Compute L_2 norm distances for a single row of $\mathbf{T}^m$
  Compute median template index and , j^*, and component threshold, τ^{m^*}, using Eq. 2 and Eq. 3
  Update component weights, $w^m, m = 1...M$, as in Eq. 1.
  $t \leftarrow t + 1$
end while

The LK-SMAT tracker uses the SMAT appearance model to identify different aspects of the target appearance and thus provide a template - the median template of the best matching model component - for use in the registration process.

There is a one-to-one association between the target aspect and the templates used for tracking, this is illustrated by the identity association matrix in figure 8. Only one template, the median of the matched component, is associated to an aspect.

Tracking is the process of registering new image data with the median template from the estimated best aspect, extracting a template from the estimated location, updating the appearance model (with the greedy algorithm), selecting the best component and hence medium template for registering with the next frame and so on.

The complete tracking procedure is detailed in Algorithm 1

6.2 LP-SMAT: Linear Predictors for Simultaneous Modelling and Tracking

The LP-SMAT tracker learns LPs specific to a particular aspect of the target object in order to continue to track through significant appearance changes. This association between aspects and LPs is achieved by an association matrix, $\mathbf{A}$, as illustrated in figure 9. Given a bank of L linear predictors and M appearance model components, the association matrix $\mathbf{A}$ has dimension $(L \times M)$. A zero value at $\mathbf{A}_{lm}$ indicates that predictor l is not associated to component m and therefore is deactivated when component m is active *i.e.* $m = m^*$. Each of the M components are associated to L/M LPs. For all the experiments, $M = 4$ and L=160 meaning 40

LPs are associated to each component and hence that 40 linear predictions are computed each frame.

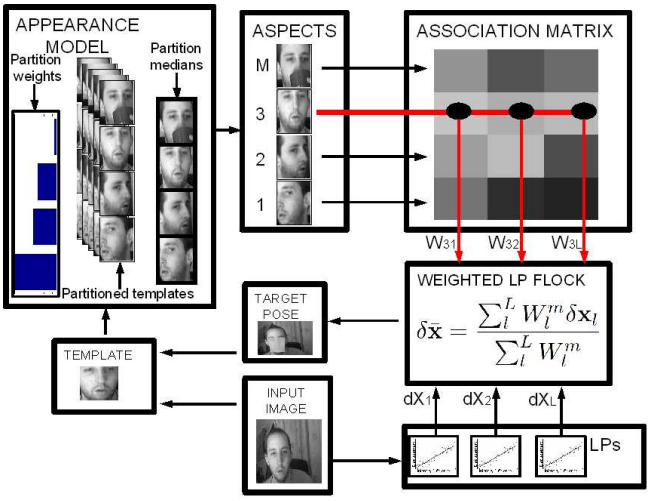


Fig. 9 *LP-SMAT system architecture*: LPs associated to the active SMAT appearance model component through association matrix are activated for tracking. The contribution each LP makes is determined by its strength of association with the current aspect. Association strengths are updated to reflect the LP's performance for the current aspect each frame.

An error function is used to continually evaluate each LP's performance over time. Rather than assigning a single error value to predictor l, error values are instead assigned to the association between each of the L predictors and each of the M appearance model components. The error values are stored in the association matrix $\mathbf{A}$ and can also be interpreted as a measure of the strength of association between a predictor and an appearance model component. The performance value used is a running average of prediction error with exponential forgetting; meaning that high values indicate poor performance. The error function used is the L_2 norm distance between predictor output $\delta\mathbf{x}_l$ and the overall tracker output $\delta\bar{\mathbf{x}}$, $\|\delta\mathbf{x}_l - \delta\bar{\mathbf{x}}\|$. Equation 15 details how the association matrix is updated with these error values. The rate of forgetting is determined by parameter β=0.1, set experimentally and unchanged in all experiments.

$$A_{lm}^{t+1} = ((1-\beta)\cdot A_{lm}^{t}) + (\beta \cdot \|\delta\mathbf{x}_l - \delta\bar{\mathbf{x}}\|) \tag{15}$$

This record of LP performance provides a method for weighting each LP's contribution to overall tracker output, $\delta\bar{\mathbf{x}}$, defined in Eq. 16 and 17.

$$W_l^m = \begin{cases} 1 - \frac{\mathbf{A}_{lm}}{\max(\mathbf{A}_{im})}, i = 1...L & \text{if } \mathbf{A}_{lm} > 0; \\ 0 & \text{if } \mathbf{A}_{lm} = 0. \end{cases} \tag{16}$$

Algorithm 2 LP-SMAT tracking procedure

$\mathbf{F}^0 \leftarrow$first image
Initialise target position $\bar{\mathbf{x}}^0$, height h and width w from user input
$M \leftarrow 4, L \leftarrow 160$
for $l = 0$ to L/M **do**
  $\mathbf{x}^l = \{rand(-h/2 : h/2), rand(-w/2 : w/2)\}$ {Randomly select reference point}
  Generate $\{\delta\mathbf{x}_i, \mathbf{d}_i\}$ {Training data}
  Compute $\mathbf{P}^l$ as is Eq. (10)
  $\mathbf{A}_{l,m=1} = 1$ {Assign all initial predictors to first mode with equal weight}
  $m^* = 1$ {Set first mode as active}
end for
while $\mathbf{F}^t \neq NULL$ **do**
  Compute $\delta\mathbf{x}^l$ as in Eq. (9) $\forall l \exists\ A_{l,m^*} > 0\ l = \{0...L\}$
  Compute $\delta\bar{\mathbf{x}}$ as in Eq. (17)
  Update predictor states $\mathbf{x}^l = \mathbf{x}^l + \delta\bar{\mathbf{x}}$
  Update association matrix, $\mathbf{A}$, as in Eq. 15
  Identify the worst predictor, ϕ, from the current active component m^* using Eq. 18.
  Extract new appearance template $\mathbf{G}^t$
  Obtain $m^* \subset \{1...M\}$ {Active component obtained by greedy assignment of new template to model component}
  Assign template $\mathbf{G}^t$ to m^* component
  Update m^* component mean and threshold as in Eq. 2 and 3.
  Learn new predictor as in Eq. (10)
  if new predictor performance $\geq$ old predictor performance **then**
    Replace worst predictor ϕ
    Update association matrix, $\mathbf{A}$, as in Eq. (15)
  end if
  $t \leftarrow t+1$
end while

$$\delta\bar{\mathbf{x}} = \frac{\sum_l^L W_l^m \delta\mathbf{x}_l}{\sum_l^L W_l^m} \tag{17}$$

A further advantage of maintaining a performance metric on each LP-aspect association is that it allows poorly performing LPs to be replaced by LPs learnt online. A new predictor is learnt for every frame from synthetic displacements of the previous frame and is evaluated on its prediction of the current frame. The worst predictor, ϕ, is identified from the current active component m^* using Eq. 18. If the prediction error of the new LP is less than the ϕ^{th} (worst) LP's error, $\|\delta\mathbf{x}_{new} - \delta\bar{\mathbf{x}}\| < \|\delta\mathbf{x}_{\phi} - \delta\bar{\mathbf{x}}\|$, then the new predictor replaces the ϕ^{th} (only in the current active component). This process serves both to introduce view-specific predictors as well as prevent outliers from contributing to the tracker output. Note that a predictor can be used by multiple components and is only completely destroyed if it has zero values for all components.

$$\phi = \underset{l}{\operatorname{argmax}}\, A_{lm^*}, l = 1...L. \tag{18}$$

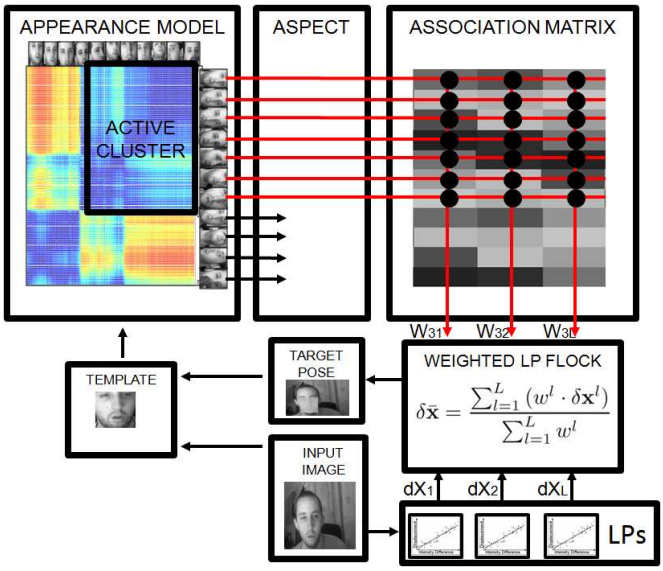


Fig. 10 *LP-MED system architecture*: The appearance templates are incrementally clustered using the medoidshift modes seeking algorithm. Each LP makes a prediction each frame and the level of contribution made is determined by its performance during each of the frames that form part of the current appearance aspect.

Note that when a new component of the appearance model is created all the predictors from the previously used component are assigned to the new component by copying a column of $\mathbf{A}$.

The complete LP-SMAT tracking algorithm is summarised in Algorithm 2.

6.3 LP-Medoidshift: Online partitioning of linear predictors for tracking

Similarly to the LP-SMAT approach, by learning aspect specific predictor weightings, each predictor can be associated to a greater or lesser extent to each aspect. However, the medoidshift clustering approach does not have a predetermined number of clusters, as in the SMAT model. The flexibility of the model is further enhanced by the possibility for appearance templates to change their cluster membership as the dataset is expanded incrementally. In order to utilise this clustering for partitioning the set of LPs, a flexible mechanism for associating clusters to LPs is required. This is achieved by maintaining a record of the performance of each LP for each template as opposed to each component in the SMAT model. A combination of template membership and these performance measures are used to compute a strength of association between each LP and any aspect.

The weighting mechanism is achieved by an association matrix, $\mathbf{A}$, as illustrated in figure 10. Given a bank

Algorithm 3 LP-Medoidshift tracking procedure

$\mathbf{F}^0 \leftarrow$first image
Initialise target position $\bar{\mathbf{x}}^0$, height h and width w from user input
for $l = 0$ to L **do**
  $\mathbf{x}^l = \{rand(-h/2 : h/2), rand(-w/2 : w/2)\}$ {Randomly select reference point}
  Generate $\{\delta\mathbf{x}_i, \mathbf{d}_i\}$ {Training data}
  Compute $\mathbf{P}^l$ as is Eq. (10)
  $w^l \leftarrow 1$ {Set all initial predictor weights to 1}
end for
while $\mathbf{F}^t \neq NULL$ **do**
  Compute $\delta\mathbf{x}^l$ as in Eq. (9) for l = {0 ... L}
  Compute $\delta\bar{\mathbf{x}}$ as in Eq. (22)
  Update predictor states $\mathbf{x}^l = \mathbf{x}^l + \delta\bar{\mathbf{x}}$
  Extract new appearance template $\mathbf{G}^t$
  Compute new row and column of distance matrix, L_2 norm $\mathbf{G}^t$ and $\{\mathbf{G}^0 ... \mathbf{G}^{t-1}\}$
  Obtain $\mathbf{T}_{i^*} \subset \{\mathbf{G}^0 ... \mathbf{G}^{t-1}\}$ {Obtained by clustering $\mathbf{T} = \{\mathbf{G}^0 ... \mathbf{G}^{t-1}\}$}
  Update association matrix, $\mathbf{A}$, as in Eq. (20)
  Identify worst predictor as in Eq. (23)
  Learn new predictor as in Eq. (10)
  if new predictor performance $\geq$ old predictor performance **then**
    Replace worst predictor l^*
    Update association matrix, $\mathbf{A}$, as in Eq. (24)
  end if
  Compute predictor weightings for next frame as in Eq. (21)
  $t \leftarrow t + 1$
end while

of L linear predictors and a set, $\mathbf{T}$, of M appearance templates, $\mathbf{T} = \{\mathbf{G}^0 ... \mathbf{G}^M\}$, the association matrix $\mathbf{A}$ has dimension $(L \times M)$. Note that M is much larger here than for the SMAT model where M indicates the number of modes rather than the number of templates. The value at $\mathbf{A}_{lm}$ indicates the strength (or weakness) of association between predictor l and template (exemplar) m. The values of $\mathbf{A}$ are set and updated using Eq. (19) and (20). Equation (19) shows how the prediction error is computed and used to initialise the association values between each predictor and the new appearance template m^t. The error is the flock agreement error, as in the LP-SMAT approach, and as detailed in section 5.4.

$$\mathbf{A}_{lm^t} = \|\delta\bar{\mathbf{x}} - \mathbf{x}_n^l\|, l = 1...L \tag{19}$$

The association values for all the other templates in the active aspect, $\mathbf{T}_{m^*} \subset \mathbf{T}$, are then updated as follows, for all predictors $l = 1...L$:

$$\mathbf{A'}_{lm} = \begin{cases} ((1-\beta)\cdot\mathbf{A}_{lm}) + (\beta \cdot \|\delta\bar{\mathbf{x}} - \mathbf{x}_n^l\|), & \text{if } \mathbf{G}^m \in \mathbf{T}_{i^*} \\ \mathbf{A}_{lm} & \text{if } \mathbf{G}^m \notin \mathbf{T}_{i^*} \end{cases} \tag{20}$$

This has the effect of smoothing the performance measures within a cluster. The values are a running average prediction error with exponential forgetting; meaning that low values of $\mathbf{A}_{lm}$ indicate greater association between predictor l and clusters containing exemplar m. As in the LP-SMAT model, the rate of forgetting is determined by parameter β=0.1, set experimentally. In all the experiments $M \leq 200$ - meaning after 200 frames, the oldest template is removed from the model - and L=80. These parameters are also set experimentally.

This error function and update strategy are used to continually evaluate predictor performance over time. This provides a means for appearance dependent weighting of each predictors contribution to overall tracker output, $\delta\bar{\mathbf{x}}$, as defined in Eq. (21) and Eq. (22).

$$w^l = 1 - \frac{\sum_{\forall m \exists \mathbf{T}_{m^*}} \mathbf{A}_{lm}}{\max \sum_{\forall m \exists \mathbf{T}_{m^*}} \mathbf{A}_{lm}} \tag{21}$$

$$\delta\bar{\mathbf{x}} = \frac{\sum_{l=1}^{L} (w^l \cdot \delta\mathbf{x}^l)}{\sum_{l=1}^{L} w^l} \tag{22}$$

The continuous evaluation of predictor performance also allows poorly performing predictors to be replaced by predictors learnt online. The worst predictor, l^*, is identified as in Eq. (23). The LP whose minimum error (over all exemplars) is greatest of all minimum errors (over all LPs) is selected.

$$l^* = \underset{\{l=1,\ldots,L\}}{\operatorname{argmax}} \left(\min_{\{m=1,\ldots,M\}} \mathbf{A}_{lm} \right) \tag{23}$$

The entries in $\mathbf{A}$ relating to the replaced predictor are updated as in Eq. (24).

$$\mathbf{A}_{l^*m} = \frac{\sum_{l=1}^{L} \mathbf{A}_{lm}}{L}, m = 1 \ldots M \tag{24}$$

The entries in $\mathbf{A}$ relating to the replaced LP are averaged across all LPs for each exemplar. The complete tracking algorithm is summarised in Algorithm 3.

7 Experiments

This section details a set of experiments used to characterise, compare and evaluate the various tracking approaches. First a convergence test is introduced and used to characterise and compare registration and regression approaches to displacement estimation as well as to investigate the effects of some of the parameters for these methods. This is followed by an experiment illustrating the benefits of the flock weighting strategy. Finally each of the tracking approaches is run on a number of challenging video sequences and the performance of each tracker is evaluated and compared.

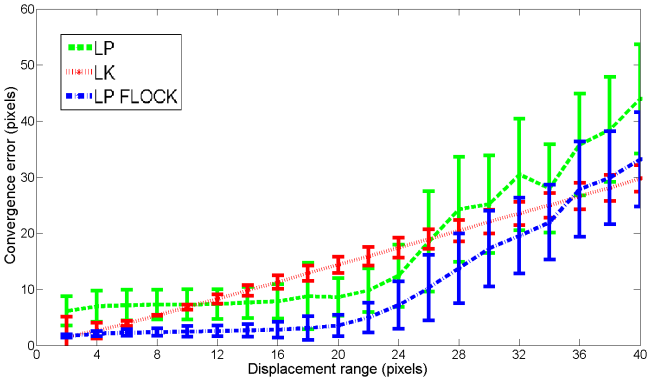


Fig. 11 Convergence error (in pixels) for three tracking approaches over a range of test displacements. The error bars represent the log of the variance of the pixel error over the 3000 tests at each range.

7.1 Convergence testing

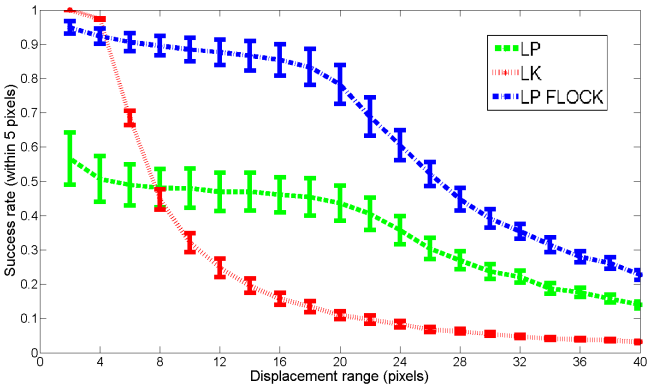


Fig. 12 Success rate of the Lucas-Kanade, LP and LP flock tracker. A test is treated as successful if the tracker converges to within 5 pixels of the true point. The error bars represent the the variance of success score over the 3000 tests at each range.

A convergence test is used to test and compare various configurations of the regression and registration tracking approaches. For registration, the test involves extracting a template at a given point, $P_{true} = \{xpos, ypos\}$, then starting the registration process at various displacements $P_{true} + d1, P_{true} + d2, ..., P_{true} + dn$, where $d = \Delta P$ and n is the number of tests carried out. The displacements can be thought of as simulated inter-frame displacements in the tracking scenario. For the regression tracking approach the test is similar - the model is learnt at P_{true} and predictions are made given

observations at displacements. The convergence test evaluates the accuracy (how close to P_{true} does the tracker get), success rate (how many tests fall within a given accuracy) and range (maximum magnitude of displacements for which tracker performs well) of the approaches.

The results represented in figures 11 and 12 are obtained by performing convergence tests using three tracking algorithms (a single LP, a flock of 60 LPs and the Lucas-Kanade registration algorithm) on a dataset of three hundred image patches (fifteen points selected on a grid from twenty images of different content, qualities and from different sources). The displacements (the horizontal axis) range from zero to forty with twenty equal steps. At each of the three hundred points, and for each of the twenty range steps, the convergence test is performed ten times giving a total of sixty thousand tests.

For the results presented in figures 11 and 12 the LP parameters are: $k = 100$ (number of support pixels), $N = 150$ (number of training examples), $r_{sp} = 20$ (support pixel range) and $r_{tr} = 20$ (training range). The LP flock is made up of 60 unweighted LPs with the same parameters. The Lucas-Kanade tracker uses the L_2 norm distance metric with a template of 20-by-20 pixels, zero order nearest neighbour interpolation and employs the Levenberg-Marquardt optimisation method.

It can be seen in figure 11 that, up to a certain range of displacements - that used in training the LP - the accuracy of both the regression methods remains fairly constant after which it degrades rapidly and linearly. The accuracy gained by the LP flock of sixty LPs is around four pixels and can be seen in figure 12 to increase the success rate by ten percent. The success rate is the proportion of tests at a given range that converge to within five pixels of the target. It is shown by the error bars in 11 and 12 that, along with accuracy, the stability of the predictions made by the LP flock is increased over the single linear prediction.

Figure 12 shows that the registration method has a greater success rate up to displacements of around five pixels, after which it degrades rapidly. This suggests the registration method has greater alignment accuracy within a certain range, the range of the basin of convergence of the alignment cost surface, than the LP flock regression approach.

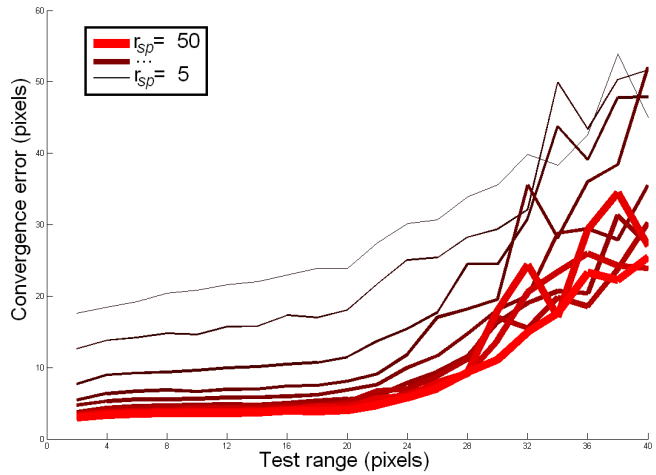
It is evident in figures 11 and 12 that the regression approaches have a greater range than the registration approach. There are methods for increasing the range of registration approaches such as image blurring and multiscale image registration (Hansen and Morse, 1999; Paquin et al, 2006). These methods essentially work by smoothing the registration cost surface thus increasing the range over which alignment can be achieved but at the cost of alignment accuracy. Performing these operations hierarchically, from course to fine, can achieve greater range and increased accuracy but with an obvious increase in computational cost. An equivalent course to fine approach has been developed for regression methods by Zimmermann et al (2009) and also Ong and Bowden (2009). The Sequential Linear Predictor (SLP) first predicts displacement using a linear regression function trained on a larger range of displacements (and hence with lower accuracy) and then with another function trained on a smaller range and so on until the required level of accuracy is obtained. The real advantage of regression techniques over registration techniques is that the range is defined by the training process as opposed to being dependent purely on the shape of the alignment cost surface *i.e.* it is possible to specify a priori the desired operating range as is explored in the following section.

7.2 Parameter effects

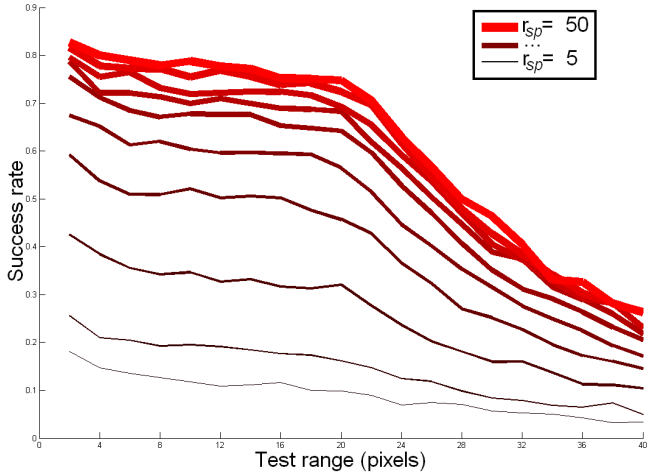
In order to evaluate the effect of various parameters on the accuracy, stability and computational cost of LP trackers, convergence tests are performed with a range of parameter configurations. The parameters explored are r_{sp} (range from reference point within which support pixels are selected), r_{tr} (range of synthetic displacements used in training), k (complexity of LP *i.e.* number of support pixels) and N (learning cost *i.e.* number of synthetic displacements used in training the LP). Rather than performing a global optimisation of these parameters (over the image dataset) these tests illustrate how the convergence characteristics of the trackers changes with varying parameters.

Figure 13(a) and 13(b) show how varying r_{sp} (the range from the reference point within which support pixels are selected) effects the LP's convergence test performance. As the support range increases, the accuracy increases. There is little or no effect on the range of displacements for which the prediction accuracy remains constant (the same as r_{tr}). Given the nature of the convergence tests (the image is static so there is no discrepancy between foreground and background) it should be noted that, in a real tracking scenario, if r_{sp} is too large it may result in the use of background pixels which would result in poor displacement predictions.

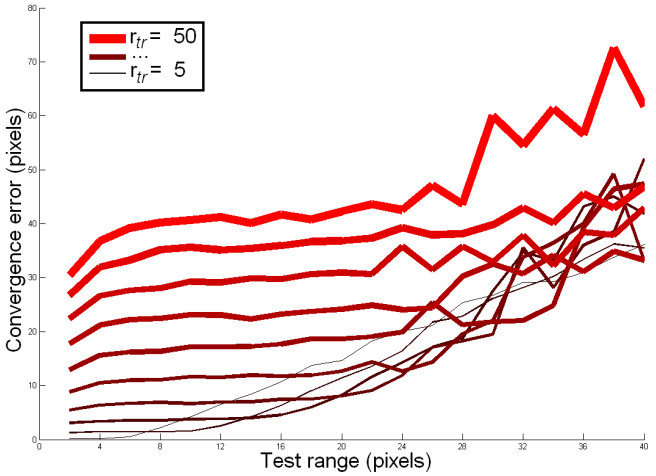
Figures 13(c) and 13(d) show how varying r_{tr} (range of synthetic displacements used in training) effects the convergence test performance. As the training range increases, the range of displacements for which the prediction accuracy remains constant also increases. This is as expected - an LP trained for displacements of up to



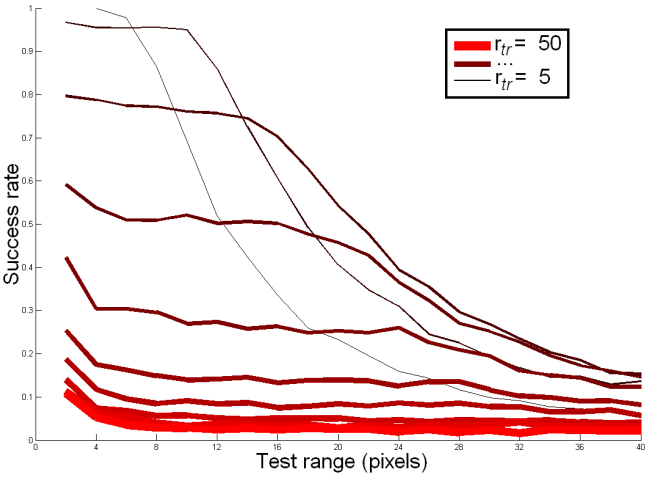
(a) Convergence error (vertical axis) with varying test ranges (horizontal axis) with different support pixel ranges, r_{sp}.



(b) Success (converges to within 5 pixels) rate (vertical axis) with varying test ranges (horizontal axis) with different support pixel range, r_{sp}.



(c) Convergence error (vertical axis) with varying test ranges (horizontal axis) with different training range, r_{tr}.



(d) Success (converges to within 5 pixels) rate (vertical axis) with varying test ranges (horizontal axis) with different training range, r_{tr}.

Fig. 13 The effect of varying the support pixel range, r_{sp}, and training range, r_{tr}, on pixel errors and success rates. Larger (redder in colour online version) lines indicate greater r_{sp}/r_{tr}. Values for r_{sp} and r_{tr} start at 5 pixels, increasing to 50 pixels in steps of 5.

20 pixels will perform consistently for test ranges of 20 pixels or less and poorly for displacements greater than 20. The trade-off for this increase in operating range is lower prediction accuracy (this result is also illustrated in figure 6).

7.3 Prediction evaluation and flock weighting

The LP flock weighting strategy introduced in section 5.4 provides a mechanism for controlling each individual LP's contribution to the overall flock output. The level of contribution an LP makes can be influenced by two factors. Firstly, due to the random selection of reference point and support pixels and the inherent weakness of the LSQ method, an LP may be consistently poor at predicting displacements. This would imply the LP should make a small contribution and that it should have a low weight. Secondly an LP's ability to predict may also be affected by changes to pixel intensities on the target. Such intensity changes may be brought about by changes to the appearance of the target, occlusions and, in the case of 2D translation LPs, out of plane displacements, rotation or affine deformation. If, for example, part of the target being tracked by an LP flock should become occluded, then those LPs whose reference point and support pixels are occluded - or indeed close to the occlusion boundary - will be con-

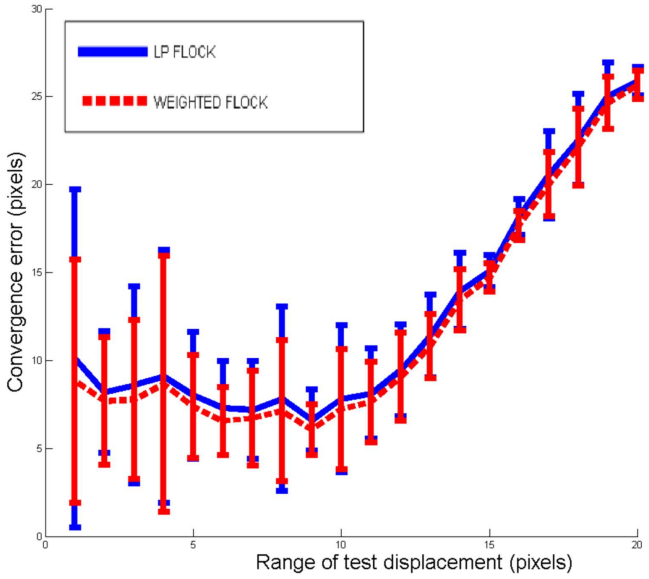


Fig. 14 Average result of sixty thousand convergence tests on weighted flocks of LPs using images with synthetic occlusions. The weighted flock is more accurate than the unweighted flock across all displacement ranges. The stability of the flock also improves slightly as can be seen by the shorter error bars.

siderably less reliable during the occlusion and hence would benefit from receiving a low weight.

In section 4, appearance models that can handle pixel intensity changes on the target, such as those brought about by occlusion, were developed and section 6 details adaptive weighting mechanisms to control the contribution of LPs given the current state of the target. To demonstrate the principle and effectiveness of the weighting mechanism presented, an experiment comparing both the weighted and unweighted LP flock is presented. Figure 14 shows the results of running the two methods on a convergence test involving deformation to the target. Sixty thousand randomly selected displacements from the 300 image patches are made and the prediction/convergence error is recorded. However, after the LP flock is learnt and for each of the 20 test ranges, a 5-by-5 pixel area randomly located within 20 pixels of the target reference point is masked *i.e.* the 25 pixels are set to white thus synthesising an occlusion on the target.

The flock agreement error, $\|\delta\bar{\mathbf{x}} - \mathbf{x}_n^p\|$, computed on the current flock output is used to re-weight contributions and hence compute the ultimate flock displacement prediction. As can be seen in figure 14, the weighted flock is consistently more accurate than the unweighted flock. The stability of the flock also improves slightly as can be seen by the shorter error bars in figure 14.

It should be noted that the results shown in figure 14 only demonstrate that the flock agreement error can be used to weight poorly performing or occluded LPs. In this experiment no LPs are replaced and no multi-modal appearance models are used. The usefulness of the weighting approach is more thoroughly demonstrated in the comparison between the LP-FLOCK and LP-SMAT/LP-MED trackers in section 7.4. For the LP-SMAT and LP-MED trackers the weighting is used to evaluate, remove and replace LPs as well as form aspect specific sets of LPs.

Table 2 Parameters settings for all methods.

Parameter	Meaning	Value
r_{sp}	Range around LP reference point within which support pixels are sampled.	20
r_{tr}	Maximum magnitude of displacements used for training LP	30
k	LP complexity: number of support pixels.	150
N	Training complexity: number of training examples.	100
L	Max. number of predictors across all modes.	160(SMAT) 80(LP-MED)
M	SMAT: Max. number of modes in model.	4
M	LP-MED: Max. number of appearance templates.	200
α	SMAT model learning rate	0.2
β	LP-SMAT and LP-MED rate of forgetting for association updates	0.1

7.4 System Evaluation

This section presents experiments evaluating the tracking performance in terms of accuracy and efficiency and provides comparison to other state of the art simultaneous modelling and tracking approaches. First each of the investigated trackers is reviewed, then the datasets used are detailed and tracking performance is evaluated. Videos demonstrating each of the trackers on the sequences are available here[3].

Trackers[4]: The trackers under investigation in this section are:

1. **LK** - the inverse compositional LK tracker using L_2 norm and Levenberg-Marquardt optimisation,
2. **LK-SMAT** - as described in section 6.1 and (Dowson and Bowden, 2006),
3. **LP-FLOCK** - as in section 5.4 with 60 LPs.
4. **LP-SMAT** - as in section 6.2,

[3] www.cvl.isy.liu.se/research/adaptive-regression-tracking

[4] Links to implementations for trackers (1) and (2) available at www.cvl.isy.liu.se/research/adaptive-regression-tracking and for (6) and (7) at www.vision.ee.ethz.ch/boostingTrackers

5. **LP-MED** - as in section 6.3,
6. **Online-Boost** - The tracker introduced by Grabner et al (2006) that tracks by online boosting discriminative foreground/background classifiers, and
7. **Semi-Online-Boost** - the online boosting tracker with a semi-supervised classifier update (Grabner et al, 2008).

Results for tracker (7) are only presented if and when tracker (6) is shown to fail where other techniques succeed. All parameters for trackers (6) and (7) are default and for all other trackers are as detailed in table 2 and no parameter tuning is performed.

Datasets[5]: The datasets used for evaluation are detailed in table 3. The **Car-Surveillance** is a benchmark sequence in the IEEE International Workshops on Performance Evaluation of Tracking and Surveillance (PETS'2000) featuring a car from a surveillance camera. The **Dudek-Face** sequence was presented in (Jepson et al, 2001) to demonstrate the trackers handling of appearance changes and un-modeled pose deformations. The **Runner** sequence is a typical track athletics sequence. The **Head-Motion** is a sequence of a moving head and torso, and the **Camera-Shake** sequence is taken from a moving webcam pointing at a phone and undergoing vigorous shaking causing motion blur and large inter-frame displacements.

Table 3 Summary of datasets

Name	Image	#frames	Introduced
Car-Surveillance		282	PETS'2000
Dudek-Face		1144	Jepson et al (2001)
Runner		400	Dowson et al (2006)
Head-Motion		2350	New
Camera-Shake		989	New

For the Car-Surveillance sequence, the target was successfully tracked by the LP-SMAT, LP-MED and Online-Boost in all 282 frames (as the Online-Boost tracker is successful the Semi-Online-Boost tracker is not tested). All other trackers fail to track the target through the appearance changes within the first 50 frames. Figure 15 shows tracking results for the first last and middle frames of the sequence. Only the LP-MED and Online-Boost trackers are marked for clarity - the LP-SMAT result is very similar to the LP-MED result in this case. Also shown in figure 15 are the four SMAT model median templates at frame 141. The current aspect median is marked with a bullseye. The LP-SMAT tracker operated at an average of 24 frames per second (fps), the LP-MED at an average 20 fps and the Online-Boost at 16 fps.

Fig. 15 Tracking results for LP-MED (solid rectangle) and Online-Boost (dashed rectangle) are shown for first, last and middle frame of sequence along with the median templates identified by the SMAT appearance model during frame 141.

The Dudek-Face sequence was not tracked entirely by any of the tracking approaches under investigation. Figure 16 highlights frames from the sequence with each trackers estimated pose marked. At around frame 155 both the LK and the LK-SMAT trackers begin to drift. Between frames 203 and 223 the hand is passed over the face causing the LP-FLOCK tracker to leave the target. The LP-SMAT, LP-MED and Online-Boost trackers are able to track through the hand occlusion. At around frame 364 the glasses are removed from the face, this causes a momentary appearance change as well as a longer term change. All the remaining three trackers cope with this appearance change. At around frame 650 the LP-SMAT tracker begins to drift, tracking only the lower part of the face, this is followed around 100 frames later by the LP-MED tracker loosing the target. Around 50 frames after the LP-SMAT tracker has lost track the Online-Boost tracker also looses track and begins to adapt to the background. By chance the face moves back into the area being tracked and the Online-Boost tracker is able to recover for a short while before losing track again for the last 10 frames. The Semi-Online-Boost tracker was also tested on this sequence, but produced poorer results than the LP-SMAT, LP-MED and Online-Boost trackers. During much of the sequence, the Semi-Online-Boost tracker produces no output and a number of false positive detections. It should be noted that (Grabner et al, 2006) report re-

[5] All datasets and ground-truth (where present) available at www.cvl.isy.liu.se/research/adaptive-regression-tracking

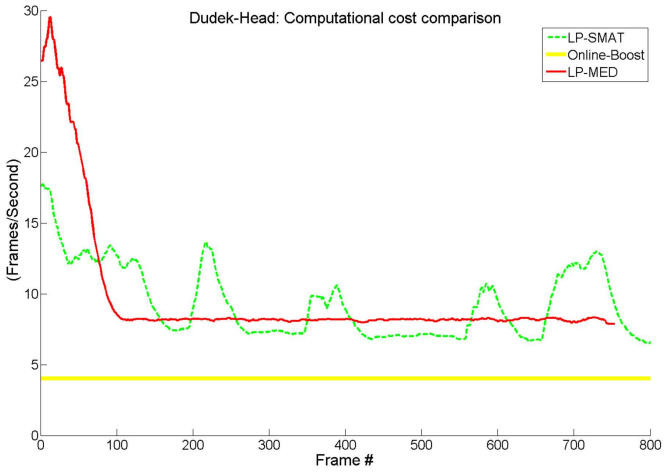


Fig. 17 The floating average frames per second is shown for LP-SMAT and LP-MED trackers is shown. The average speed of the Online-Boost tracker on this sequence is shown for reference.

Fig. 18 SMAT model medians for four key frames. The medians are sorted with decreasing weight left to right. The four key frames are: during and after the hand occlusion, and during and after the removal of the glasses.

sults showing another version of the Online-Boost boost tracker successfully tracking the Dudek-Face sequence, however these results are not achievable with the simpler implementation that is made publicly available.

The tracked region in the Dudek-Face sequence is 130x130 pixels. The Online-Boost tracker runs at a fairly constant 4 fps and the LP-SMAT and LP-MED trackers run at an average 8 fps and 10 fps respectively. Figure 17 compares the computational cost of these three methods. As the frame by frame processing time is not available for the Online-Boost tracker just the average frames per second is plotted. The Online-Boost algorithm has a very consistent computation time per frame. From this figure it can be seen that the LP-MED tracker also operates at a stable speed after an initial period. This initial high frame rate is due to having few examples in memory and hence a small distance matrix and association matrix. Interestingly, the occasional peaks in the LP-SMAT frame rate (seen in figure 17) coincide with significant events in the sequence *e.g.* the first and second peaks (at around 200 frames and 350 frames) coincide with the hand passing over the face and with the glasses being removed respectively. This is due to the creation of new modes during these transient appearance changes. As a new mode will be represented by very few templates the cost of maintaining the distance matrix between each template is low. As the component becomes populated with new templates the cost of maintaining the distance matrix rises again as is shown by the falling frame rate after each event.

Figure 18 shows the SMAT model medians for four key frames. The medians are sorted with decreasing weight left to right. The four key frames are: during and after the hand occlusion, and during and after the removal of the glasses. It can be seen that the SMAT model quickly builds a new mode to represent each of the transient changes of appearance. After the hand passes away from the face the appearance returns to an earlier aspect and so the previously learnt predictor weightings are re-employed. After the glasses have been removed a new mode is created to represent the new appearance and this mode soon has the highest weight *i.e.* most resembles the estimated target appearance.

The Runner sequence features athletes running through the bend of a race track and then down the straight towards the camera. The trackers are initialised in the first frame on the only athlete to remain in the scene for the whole sequence. As can be seen in figure 19, the Online-Boost tracker (yellow dashed rectangle) jumps up to start tracking the head and upper torso of the athlete at frame 10 whereas all the other trackers stay tracking the central torso area. This is reflected in the positional accuracy graph in figure 20. As the ground-truth position (obtained by hand labeling) is centered on the athlete the Online-Boost tracker accumulates greater area once it starts to track the head of the athlete. Result for the Semi-Online-Boost tracker are not shown here as the tracker fails early on and, due to the significant appearance changes early in the sequence, does not produce any output for most of the sequence. The LK tracker (blue rectangle) begins to drift and loses track completely by frame 150 (refer to figures 19 and 20). The LP-FLOCK tracker begins to lose track at around frame 100. The Online-Boost, LK-SMAT, LP-SMAT and LP-MED continue to track through significant appearance changes (even though the Online-Boost is tracking a different part of the ath-

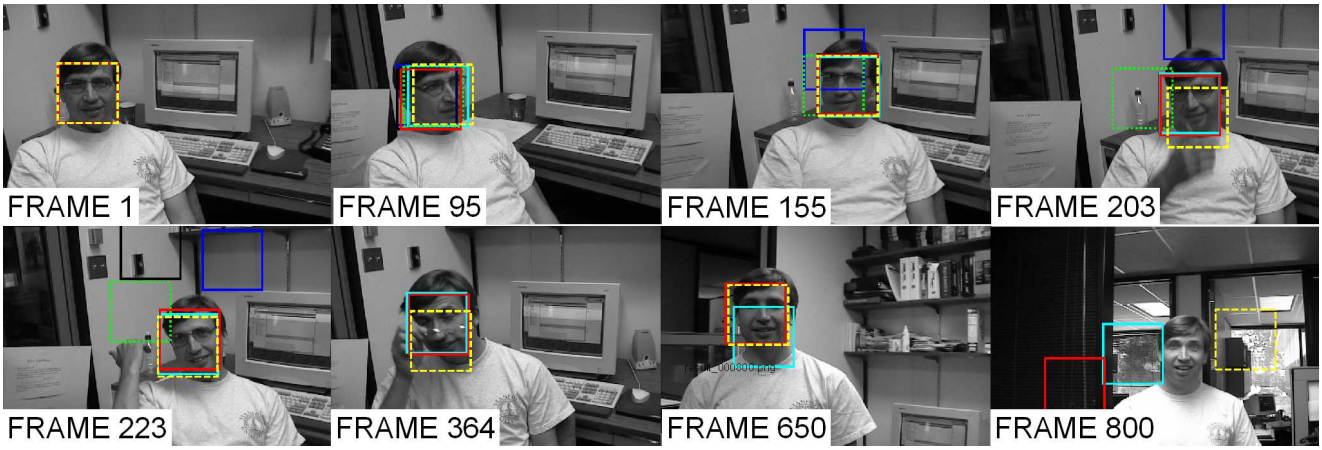


Fig. 16 Highlighted frames from the Dudek-Face sequence. Tracker key: Dark blue - LK, black - LP-FLOCK, green - LK-SMAT, light blue - LP-SMAT, red - LP-MED, yellow - Online-Boost. Colour online.

Table 4 Average frame rate per second. Template sizes for each sequence are: CAR - 40x20, DUDEK - 130x130, RUNNER - 38x126, HEAD - 90x100, CAM-SHAKE - 25x25. Parameters for all methods are unchanged for each sequence and set as detailed in the relevant part of section 6.

Tracker	CAR	DUDEK	RUNNER	HEAD	SHAKE
LK-SMAT	12	2	6	6	12
LP-FLOCK	65	16	24	25	65
LP-SMAT	24	10	15	16	35
LP-MED	20	8	12	12	20
Online-Boost	16	4	7	7	16

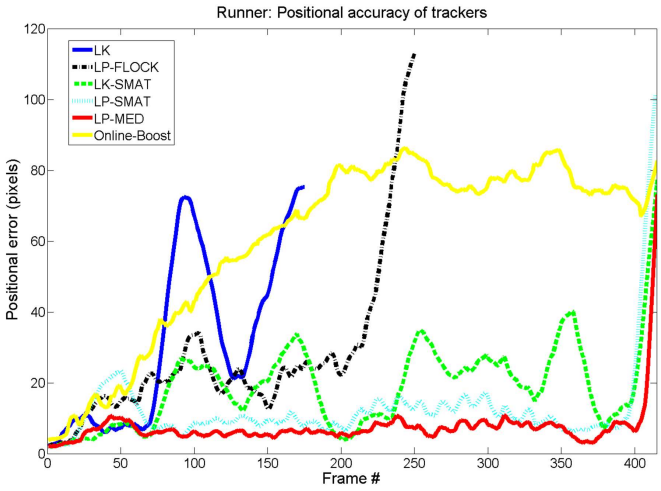


Fig. 20 The positional error for each tracker is shown. Ground-truth was obtained by hand labeling the sequence.

lete it does not loose track until around frame 410). The LP-MED tracker stays on the target longer than all the other trackers and gives a more stable track of the target. All trackers fail by frame 420. The frame rates for each tracker on this and all other sequences are given in table 4.

The head tracking sequence consists of 2500 frames with the head undergoing large pose variations and at one point becoming occluded by a cup for over 100 frames. The LK-SMAT, LP-SMAT, LP-MED, Online-Boost and Semi-Online-Boost trackers each track the head throughout the whole sequence but the methods with no online appearance learning (LK and LP-FLOCK) both fail to track the target. Figure 21 shows the head being tracked by the LP-MED tracker (top row). On the bottom row of figure 21, the position of the LPs are indicated (black spot) as well as the support pixel range, r_{sp}, (white dashed circles). Also marked on the bottom row are the positions of the worst predictor from the current frame (red mark) and the predictor learnt in the current frame (green mark). As can be seen the worst predictors often lie with most or all the support pixels on the background or, in the case of the cup, on the occluding object. These predictors are not necessarily removed, they may be re-employed later in the sequence when a previous aspect is presented.

The Camera-Shake sequence is captured from a low cost web cam and is of a static scene and a moving camera. The camera undergoes considerable shaking causing large inter frame displacements as well as translation, rotation and tilting. Figure 22 shows results on this sequence for the LP-MED (red), Online-Boost (yellow solid) and Semi-Online-Boost (yellow dashed) trackers (results for the LP-SMAT tracker are similar to the LP-MED tracker on this sequence, though a little less accurate). All trackers except LP-SMAT and LP-MED fail on this sequence by frame 280 (the onset of some camera shake). The Semi-Online-Boost algorithm is able to re-initialise a number of times during the sequence but is never able to track while the camera is being shaken due to high inter-frame displacements and image blur. The displacement predicted from frame 374 to 375 is 37 pixels (16 vertical and 33 horizontal) and despite the significant blurring in frame 375, the

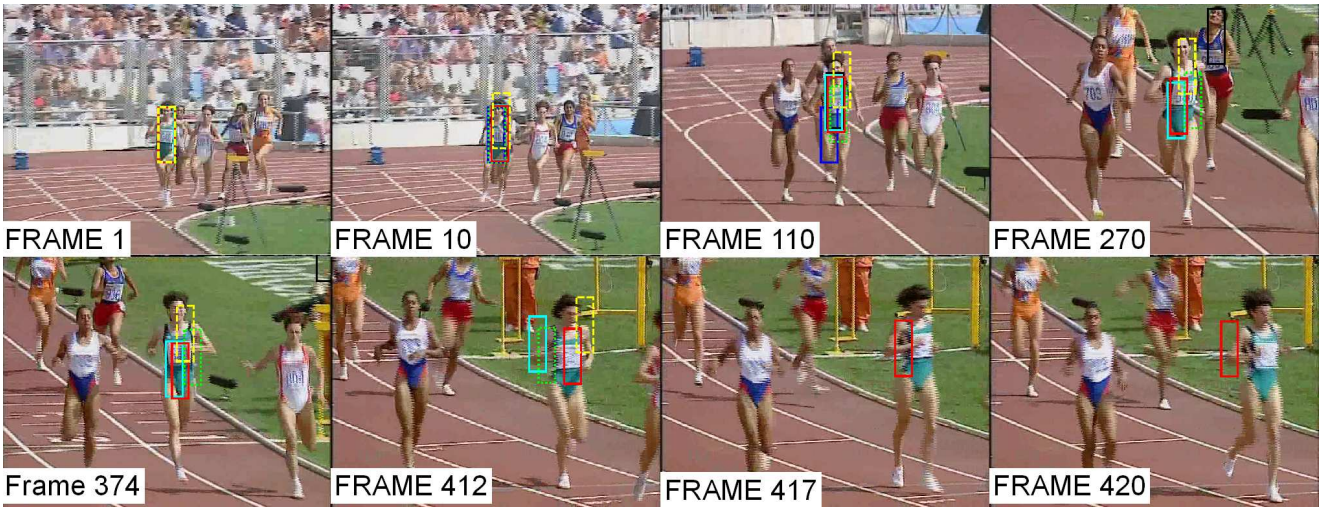


Fig. 19 Highlighted frames from the track running sequence. Tracker key: Dark blue - LK, black - LP-FLOCK, green - LK-SMAT, light blue - LP-SMAT, red - LP-MED, yellow - Online-Boost. Colour online.

Fig. 21 The medoidshift algorithm tracks the head sequence (top row). LP positions (small black dots) and support pixel ranges (white dashed circles) are shown as well as the worst predictor from the current frame (large black circle) and the predictor learnt from this frame (large white circle). It is this process that generates view-specific displacement predictors within the model.

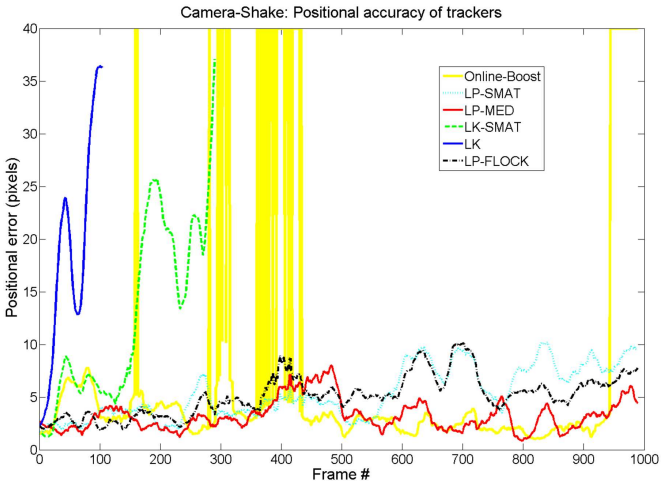


Fig. 23 The positional error for each tracker is shown. Where the Online-Boost tracker fails and is re-initialised is indicated by the vertical yellow lines.

tracker still succeeds in making a low error prediction. Due to online learning of predictors, some are learnt from blurred images allowing for prediction during this blur. Figure 23 shows the positional accuracy plots generated by the six trackers on this sequence. Due to the limited basin of convergence both the alignment based trackers fail to deal with the large inter frame displacements and SMAT loses track as soon as the camera starts to shake. On this sequence the LP-Medoidshift approach achieves more accurate results than the other two LP trackers.

For all sequences the target patch is identified by hand only in the first frame, all algorithm parameters are unchanged between sequences. Ground truth for every frame of the athletics and camera motion sequences was achieved by hand labeling and was used to generate the error plots in figures 23 and 20.

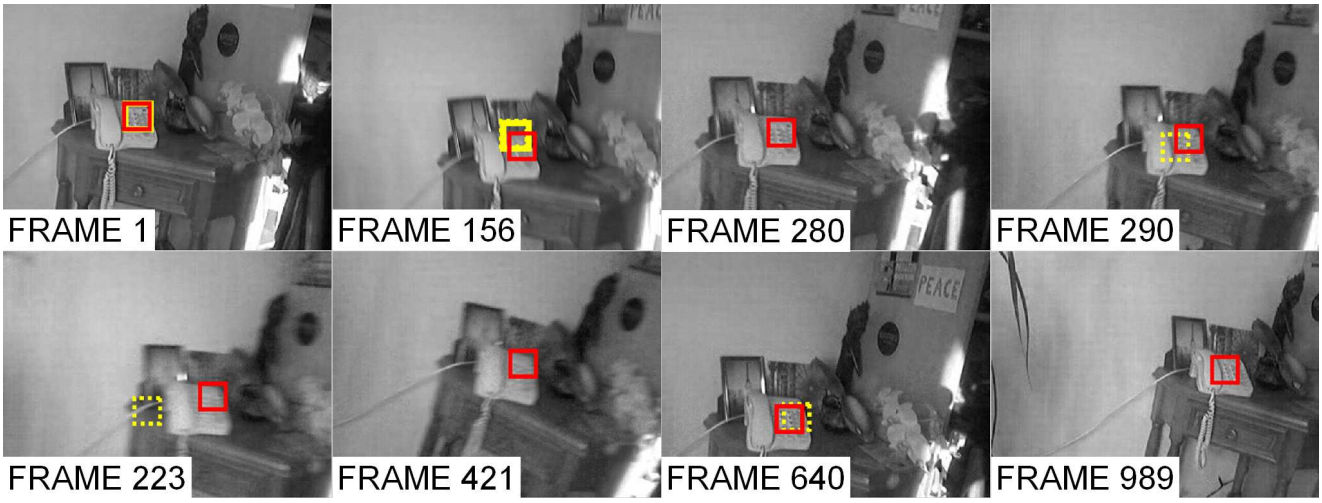


Fig. 22 Highlighted frames from the shaking camera sequence. Tracker key: Solid dark grey (red) - LP-MED, solid white (yellow) - Online-Boost, dashed white (yellow) Semi-Online-Boost.

8 Conclusion

An online Linear Predictor tracker has been introduced and has been shown to be applicable to a number of tracking scenarios. The online LP allows for the explicit trade-off between predictor range, accuracy and computational cost as shown by the convergence tests in section 7.1. Essential to the use of the online-LP are mechanisms to evaluate, weight, remove and relearn LPs online during tracking. A general tracking framework that combines online-LP flocks with methods for online appearance model learning has been developed and various configurations investigated. Within this framework the tracking process provides a mechanism for self supervision of the appearance model learning process and, in return, the appearance model provides information about the structure of the target appearance space that enables the tracker to cope with a high degree of variation in appearance. The only supervision required is to identify the location of the target in the first frame.

Two methods for appearance model learning are introduced. The SMAT model partitions the appearance space into a predefined number modes that represent aspects of the target. While this does limit the flexibility of the model and may lead to potential misrepresentation of the underlying appearance distribution, the SMAT tracker has demonstrated the ability to adapt to large appearance changes very quickly. The ability to introduce new modes on-the-fly to represent transient target appearance changes (*e.g.* occlusion of face by hand, see figure 18), whilst maintaining unchanged the representation of other parts of the appearance space, proves highly beneficial for tracking many objects.

The medoidshift algorithm on the other hand benefits from the increased flexibility and hence greater likelihood to build more representative models of the object appearance space, regardless of the temporal evolution of the target appearance. This flexibility also extends into the general tracking framework by allowing a more flexible association strategy between aspects of the target and the online-LPs.

The tracking approaches investigated in this work model 2D target transformations (translation). Any higher order transformations of the target (*e.g.* scale or rotation) are considered as appearance changes, and treated as new aspects within the adaptive appearance models. Whilst the introduction of higher order transformations is possible, and may considerably improve results in some cases (*e.g.* the Dudek-Face and Runner sequence, in which the target undergoes considerable scale changes), it has been found that the introduction of more parameters into the warping function can increase the risk of drift and the number of local minima (Dowson and Bowden, 2006).

An objective of this work is to delineate the class of problems for which the proposed methods are preferential by including examples of partial failure (Dudek-Face) and examples where this method outperforms the other approaches (Camera-Shake). The success of the LP based techniques on the Camera-Shake sequence highlight the ability of regression techniques to cope with large inter-frame displacements in the presence of local minima. On the other hand, registration techniques tend to achieve higher accuracy provided the displacement is within the basin of convergence.

Other tracking approaches ((Ross et al, 2008; Jepson et al, 2001)) have been shown to track the Dudek-Face sequence. Compared to the proposed approach, these methods are slower, but handle the significant appearance changes in this sequence more robustly. Given

that these are affine trackers and the trackers compared here only model translation, direct comparison is not included as this would raise issues regarding the DoF used in tracking that are not addressed here.

Both the appearance models developed have demonstrated the ability to adapt to large variations in appearance in order to manage flocks of online-LPs and to facilitate accurate and efficient tracking. When compared to the online boosting methods, there are clearly cases where the discriminative classifiers will be better able to represent the target, but for a class of tracking problems the LP-MED/LP-SMAT model provides comparable accuracy with an increase in computational efficiency.

Due to the computational efficiency of the online-LP, the tracker is able to track targets in real time (35/20 frames per second for LP-SMAT/LP-MED), even whilst building and maintaining the appearance model. It is shown that the approach can handle large inter frame displacements and adapt to significant changes in the target appearance with low computational cost. The online-LP tracker has been shown to be particularly effective at tracking through considerable camera shake.

The advantages of such a simultaneous modelling and tracking approach are clear when considering how much hand crafting, offline learning, hand labeling and parameter tuning must be done in order to employ many existing object tracking approaches. By developing the online-LP and mechanisms to manage LP-flocks, the class of applicable tracking problems has been extended to included, amongst others, automatic seeding of the tracker. For example, a computer vision application may seed a target tracker - such as LP-SMAT or LP-MED - on a region of detected motion, without any prior on target appearance.

Many applications require tracking that operates at high frame rates and can handle high object velocities as well as be robust to significant appearance changes and occlusion. This is achieved here by utilising the computationally efficient technique of least squares prediction and online target appearance modelling.

Acknowledgements The authors would like to thank Karel Zimmermann for his useful discussions regarding regression tracking and Helmut Grabner for providing advice regarding the Online-Boost tracker.

References

Avidan S (2007) Ensemble tracking. IEEE Transactions on Pattern Analysis and Machine Intelligence 29(2):261–271

Baker S, Matthews I (2004) Lucas-kanade 20 years on: A unifying framework. International Journal of Computer Vision 56(3):221–255

Bray M, Kohli P, Torr PHS (2006) Posecut: Simultaneous segmentation and 3d pose estimation of humans using dynamic graph-cuts. In: ECCV (2), pp 642–655

Collins R, Liu Y, Leordeanu M (2005) On-line selection of discriminative tracking features. IEEE Transaction on Pattern Analysis and Machine Intelligence 27(1):1631 – 1643

Cootes TF, Edwards GJ, Taylor CJ (1998) Active appearance models. In: ECCV (2), pp 484–498

Dowson N, Bowden R (2006) N-tier simultaneous modelling and tracking for arbitrary warps. p II:569

Dowson N, Bowden R (2008) Mutual information for lucas-kanade tracking (milk): An inverse compositional formulation. IEEE Transactions on Pattern Analysis and Machine Intelligence 30(1):180–185

Ellis L, Matas J, Bowden R (2008) Online learning and partitioning of linear displacement predictors for tracking. In: BMVC (1), pp 33–42

Grabner H, Grabner M, Bischof H (2006) Real-time tracking via on-line boosting. In: BMVC06, British Machine Vision Association, pp 47–56

Grabner H, Leistner C, Bischof H (2008) Semi-supervised on-line boosting for robust tracking. In: ECCV '08: Proceedings of the 10th European Conference on Computer Vision, Springer-Verlag, Berlin, Heidelberg, pp 234–247

Hager GD, Belhumeur PN (1998) Efficient region tracking with parametric models of geometry and illumination. IEEE Transactions on Pattern Analysis and Machine Intelligence 20(10):1025–1039

Hansen BB, Morse BS (1999) Multiscale image registration using scale trace correlation. Computer Vision and Pattern Recognition, IEEE Computer Society Conference on 2:2202

Isard M, Blake A (1998) Condensation - conditional density propagation for visual tracking. International Journal of Computer Vision 29:5–28

Jepson AD, Fleet DJ, El-Maraghi TF (2001) Robust online appearance models for visual tracking. In: CVPR (1), pp 415–422

Jurie F, Dhome M (2002) Real time robust template matching. In: in British Machine Vision Conference 2002, pp 123–131

Kalal Z, Matas J, Mikolajczyk K (2010) P-N Learning: Bootstrapping Binary Classifiers by Structural Constraints. CVPR

Lucas BD, Kanade T (1981) An iterative image registration technique with an application to stereo vision. In: IJCAI, pp 674–679

Marchand É, Bouthemy P, Chaumette F, Moreau V (1999) Robust real-time visual tracking using a 2d-3d model-based approach. In: ICCV, pp 262–268

Marquardt DW (1963) An algorithm for least-squares estimation of nonlinear parameters. SIAM Journal on Applied Mathematics 11(2):431–441

Matas J, Zimmermann K, Svoboda T, Hilton A (2006) Learning efficient linear predictors for motion estimation. In: ICVGIP, pp 445–456

Matthews I, Ishikawa T, Baker S (2004) The template update problem. IEEE Transactions on Pattern Analysis and Machine Intelligence 26(6):810–815

Mayol WW, Murray DW (2008) Tracking with general regression. Mach Vision Appl 19(1):65–72

Ong EJ, Bowden R (2009) Robust facial feature tracking using selected multi-resolution linear predictors. pp 1483–1490..

Paquin D, Levy D, Schreibmann E, Xing L (2006) Multiscale image registration. Math Biosci Eng 3(2):389–418

Ross DA, Lim J, Lin RS, Yang MH (2008) Incremental learning for robust visual tracking. International Journal of Computer Vision 77(1-3):125–141

Sheikh YA, Khan EA, Kanade T (2007) Mode-seeking by medoidshifts. In: Eleventh IEEE International Conference on Computer Vision (ICCV 2007), 141

Shum HY, Szeliski R (2000) Systems and experiment paper: Construction of panoramic image mosaics with global and local alignment. International Journal of Computer Vision 36(2):101–130

Viola P, Jones M (2002) Robust real-time object detection. International Journal of Computer Vision 57(2):137–154

Williams OMC, Blake A, Cipolla R (2003) A sparse probabilistic learning algorithm for real-time tracking. In: ICCV, pp 353–361

Zimmermann K (2008) Fast learnable methods for object tracking. PhD thesis, Center for Machine Perception (CMP), Czech Technical University, Prague, Czech Republic

Zimmermann K, Matas J, Svoboda T (2009) Tracking by an optimal sequence of linear predictors. IEEE Transactions on Pattern Analysis and Machine Intelligence 31(4):677–692