FAST VISUAL TRACKING BY TEMPORAL CONSENSUS

A. H. Gee and R. Cipolla

CUED/F-INFENG/TR 207

February 1995

University of Cambridge
Department of Engineering
Trumpington Street
Cambridge CB2 1PZ
England

Email: ahg/cipolla @eng.cam.ac.uk

Fast Visual Tracking By Temporal Consensus

Andrew Gee and Roberto Cipolla

University of Cambridge
Department of Engineering
Trumpington Street
Cambridge CB2 1PZ
England

February 1995

Abstract

At the heart of every model-based visual tracker lies a pose estimation routine. Recent work has emphasised the use of least-squares techniques which employ *all* the available data to estimate the pose. Such techniques are, however, susceptible to the sort of rogue measurements produced by visual feature detectors, often resulting in an unrecoverable tracking failure. This paper investigates an alternative approach, where a *minimal* subset of the data provides the pose estimate, and a robust regression scheme selects the best subset. Bayesian inference in the regression stage reconciles measurements taken in one frame with predictions from previous frames, eliminating the need to further filter the pose estimates. The resulting tracker performs very well on the difficult task of tracking a human face, even when the face is partially occluded. Since the tracker is tolerant of noisy, computationally cheap feature detectors, frame-rate operation is comfortably achieved on standard hardware.

Keywords: Visual tracking, pose estimation, robust regression, Bayesian inference, model acquisition.

1 Introduction and outline

A visual tracking algorithm can be found at the heart of many real-time computer vision systems. This paper is concerned with the so-called "model-based trackers", which are used to track familiar objects as they move within the field of view. While model-based trackers can be designed to cope with articulated objects [11], here we consider only rigid objects, which feature in the vast majority of applications, ranging from the visual control of robotic manipulators [8] to hands-off interfaces for human-computer interaction [2].

Almost all model-based trackers work on the same basic principle:

Measurement: A number of key features, usually corners or edges, are detected in each frame of the image sequence. These features are known to correspond to features in a 3D model of the target.

Pose estimation: The image positions of the detected features are used to calculate the target's *pose* relative to the camera: that is, its relative position and orientation.

Filter: The result of this pose calculation is often filtered to reduce the effects of noise in the feature detectors, usually drawing on some sort of smooth motion assumption (which is reasonable, given that the motion of rigid objects is governed by inertia).

Predict: The position of the target is predicted in the next frame, again using the smooth motion assumption, and the key features are searched for in small windows around their expected image positions. This *local* search allows the tracker to run quickly, and overcomes any problems establishing correspondence between model and image features.

The pose estimation problem is typically over-constrained, in that there are usually more measurements available than are required. Recent tracking algorithms [6, 7, 11] have responded to this redundancy by employing *all* the measurements and some sort of least-squares pose fitting criterion. The problem with such approaches is that they are easily deceived by the sort of rogue measurements so often produced by visual feature detectors. Moreover, a poor pose estimate in one frame can easily lead to an unrecoverable tracking failure in the next.

An alternative approach is to use a minimal subset of the data to estimate the pose. So long as one subset of good, accurate measurements exists, the rest of the data can be ignored and gross errors will have no effect on the tracking performance. The key to the design of such trackers lies in the selection of the best subset. Typically, some sort of clustering or consensus criterion is used: if the pose suggested by one subset also explains a significant proportion of the other measurements, then that pose is very likely to be correct. While some trackers have already been developed along these lines, they have tended to be rather specialized, requiring for instance stereo camera set-ups [15] or restrictive vertex-pair features [16]. Here we present a general tracking algorithm, which works with a single camera and is suited to a wide range of difficult targets. State-of-the art performance is achieved by properly reconciling single frame measurements with inter-frame motion constraints.

In Section 2 we discuss the problems with least-squares pose fitting and go on to consider alternative, minimal information techniques. The "alignment" algorithm [10] is reviewed as a robust technique for estimating the pose from just three image points. By placing the alignment algorithm within the RANSAC regression paradigm [4], it is possible to reliably select the best three points by looking for a consensus with other points in the image. In Section 3 we consider the use of RANSAC/alignment pose estimation in a visual tracking algorithm. The question here is how to reconcile the pose estimate from a single frame with predictions from previous frames to produce the best tracking performance. While this could be tackled using a separate Kalman filter, a more elegant solution would be to perform the filtering in conjunction with the pose estimation. This is achieved within a Bayesian inference framework, which describes how to properly combine intra-frame consensus information with inter-frame motion constraints. The resulting algorithm is applied to the difficult task of tracking a human face under adverse conditions: low contrast, rapid motion and occasional occlusion. The performance of the tracker is very good, and its robustness permits the use of noisy, computationally cheap feature detectors, so that frame-rate operation is comfortably achieved on standard hardware.

2 Regression techniques for pose estimation

Least-squares

Suppose we have a set of points in an image with known correspondences with a set of points in a full 3D model of an object, and we wish to calculate the pose of the object relative to the camera. Even though the problem can be solved using only a few of

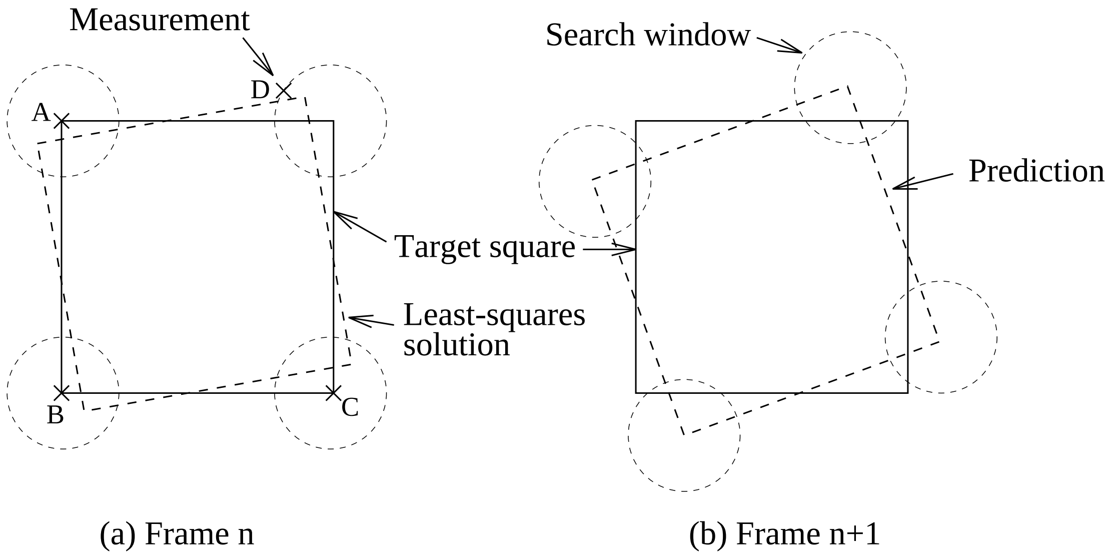


Figure 1: Least-squares tracking.

Consider the task of visually tracking a square in the plane, by detecting its corners in each frame. The square is free to translate and rotate within the plane, and isotropic expansion is also allowed, though the square is, in fact, stationary. In frame n a gross error from one corner detector produces a least squares solution which implies a 10° rotation. If a constant velocity model is used to predict the square's location in frame n+1, then it is entirely possible that all the actual corners might lie outside the local search windows used by the feature detectors, leading to a serious tracking failure.

the points, it is intuitively sensible to try to use all the available information, as this should produce the most accurate result. The resulting over-constrained problem could be tackled using some sort of least-squares technique, perhaps weighting each point's influence by the amount of confidence we have in that particular measurement. Indeed, a least-squares approach is often taken in object recognition and tracking systems: see, for example, [6, 7, 11].

The problem with least-squares techniques is that they are easily deceived by a rogue measurement. Suppose that one of the point detectors produced a gross error (which is bound to happen eventually with simple feature detectors), as in Figure 1(a). The least-squares solution for the object's pose is clearly poor. More serious, though, is the effect one poor result can have on the subsequent tracking performance. If the least-squares solution was used to predict the object's position in the next frame, then it is entirely possible that all the actual points might lie outside the search windows of the point detectors, leading to a catastrophic tracking failure, as in Figure 1(b).

Minimal information techniques

At the other end of the spectrum are those techniques which attempt to calculate the pose using the minimum amount of information. For example, we could calculate the pose of the square in Figure 1(a) using only the corners detected at A and B, which would give

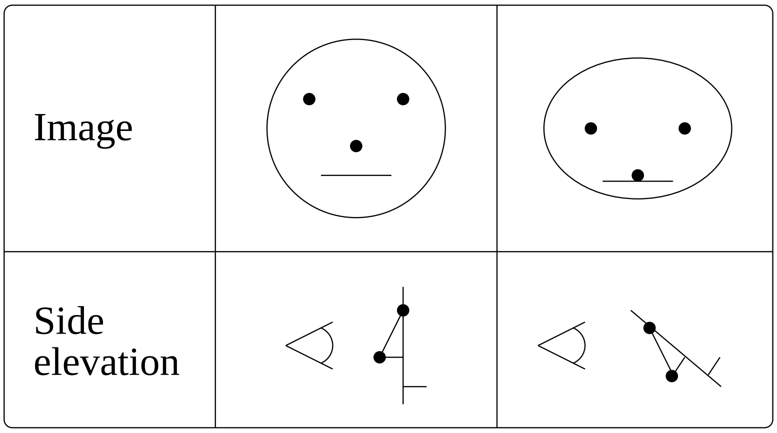


Figure 2: Three points admit two poses under weak-perspective.

The figure shows a stylized face in two different poses being viewed through a weak-perspective camera. In both images the eye and nose points appear in identical relative position. The alignment algorithm could be used to infer the two poses from the eye and nose points.

a perfect solution. For the case of three-dimensional objects viewed through a camera, the smallest number of points which usefully constrain the pose is three. Assuming a full-perspective imaging model, equivalent to a pinhole camera, a closed-form technique exists to calculate the pose, up to a four-fold ambiguity, from three image points [4]. However, the calculation requires significant camera calibration, and is also very complicated.

An alternative is to assume a weak-perspective (or scaled orthographic) imaging process, valid when the viewing distance is large compared with the depth variation across the target. Since weak-perspective pose estimation is used extensively in the rest of this paper, we shall take some time here to examine it in a little detail. The weak-perspective projection from a model point (X, Y, Z) in model-centered coordinates to an image point (x, y) is described mathematically as follows:

$$\begin{bmatrix} x \\ y \end{bmatrix} = s \begin{bmatrix} R_{11} & R_{12} & R_{13} & t_x \\ R_{21} & R_{22} & R_{23} & t_y \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \tag{1}$$

The projection describes a rigid body rotation of the object by the rotation matrix $\mathbf{R}$, a translation parallel to the image plane (t_x, t_y), orthographic projection onto the image plane and an isotropic scaling s in the plane. Under these viewing conditions, three points admit only two possible poses, which can be found using the "alignment" algorithm [10]. The calculation is reliable, simple, stable and closed-form. It takes as input three image points (x_i, y_i) with corresponding model points (X_i, Y_i, Z_i), and returns the rotation matrix $\mathbf{R}$, the translation vector (t_x, t_y) and the scale factor s for the two possible solutions — see Figure 2. Moreover, the calculation can be adapted to work with a combination of points and lines [14].

Corners	Consensus set
A B	A B C
A C	A B C
A D	A D
B C	A B C
B D	B D
C D	C D

Table 1: RANSAC for the square in Figure 1.

Using RANSAC, no extended consensus is found whenever the corner D is used in the pose calculation: only the two corners used to find the pose lie in the consensus set. The measurement at D is therefore rejected as an outlier, and the correct pose is found using any two of the measurements taken at corners A, B and C.

Random sample consensus

With a minimal information technique, such as the alignment algorithm, we have to decide which set of three points to use. This can be accomplished within the RANSAC (RAndom SAmple Consensus) regression paradigm [4], introduced into the vision literature in 1981. The idea behind RANSAC is to find a significant group of points which are all consistent with a particular pose, and reject the remaining points as outliers. A RANSAC/alignment-type pose calculation might take the following form:

1. Choose three image points and calculate the two solutions for the object's pose using the alignment technique.
2. Using each calculated pose, back-project the whole model into the image using (1) and identify which of the observed points lie within some error margin, say r pixels, of the back-projected points. This defines the consensus set for each candidate pose. The size of the consensus set will be greater than or equal to three.
3. Repeat 1 and 2 for all combinations of three points. Choose the pose which is supported by the largest consensus set. If there is a tie, select from the tied solutions the pose which best explains the measurements in a least-squares sense.

For the simple two-dimensional example in Figure 1(a), assuming a suitable value for r, RANSAC would produce the results in Table 1. The measurement at corner D is correctly identified as an outlier, and the pose of the square is calculated accurately using the other corners.

The single parameter in the RANSAC algorithm, r, is best determined empirically. Too small an r will result in no extended consensus being found, too large and all the candidate pose solutions will be supported by large consensus sets. r should be set somewhere between these two extremes, so that typical noisy data produces consensus sets of differing sizes. For the tracking applications described in this paper, a value of 5 pixels was used throughout. See [4] for a more thorough discussion of how r may be sensibly established using either analytical or empirical means.

The original formulation of the RANSAC paradigm differs from the process outlined above in two ways. First, we choose to consider *all* combinations of three points, whereas in the original paradigm 3-point samples were selected randomly until a suitably large consensus set was found. It would be nice to avoid this random sampling if at all possible, since it introduces another parameter into the algorithm, the size of a suitably large consensus set. With models of up to nine points, it is possible to examine all 84 of the 3-point combinations and maintain tracking at frame rate on standard hardware (see Section 3). Random sampling need be considered only for larger models, where it could be used to maintain fast tracking. The original RANSAC paradigm also called for the pose to be refined (probably using least-squares) on the evidence of the entire consensus set, not just the original three points. While this might produce higher quality results in a single image, we consider it both time-consuming and unnecessary (least-squares techniques tend to be iterative and also require a large consensus set for stability [9]).

It was pointed out in [13] that the robustness of RANSAC is contingent on the existence of at least one subset of the points being consistent with the correct pose. If Gaussian noise were to corrupt all the data, then the best pose estimate would be available from a least-squares technique, and not RANSAC. However, the Gaussian noise assumption is rarely applicable to visual feature detectors, which are prone to lock on to the wrong feature and produce gross errors. This claim is substantiated in Figure 3, which shows the measurement error as a "cheap and nasty" feature detector (just a local dark-spot finder) attempts to follow a nostril on a moving face. It is clear that this noise is not zero-mean Gaussian: the nostril is well tracked at times, and then effectively lost when the head is turned and the nostril is partially obscured.

3 Tracking using RANSAC and alignment

A consensus tracker

The RANSAC/alignment pose estimation technique is robust and fast enough to form the core of a real-time visual tracking system. We implemented a RANSAC/alignment tracker and used it to track a face using six point features and simple, computationally cheap feature detectors — see Figure 4. The 3D model coordinates of the eyes, lip shadow and eyebrow centre were measured by hand on the face: the remaining model coordinates were automatically acquired from an image sequence using the scheme outlined in Appendix A.

The output of the simple feature detectors is extremely noisy and prone to gross errors, though RANSAC regression should cope with this. The advantage of simple feature detectors is their speed: the complete six-point tracker runs at 125Hz on a Sun SparcStation 10 equipped with a frame-grabbing board[1]. At first no filtering was applied to the pose estimates, and at such high sampling rates no real prediction stage is required: the back-projected feature locations in one frame were used to initialize the feature detectors in the next.

So that different tracking algorithms could be fairly compared, a 30 second image sequence was recorded on video tape and repeatedly used for the tracking experiments. The video shows one of the authors moving his head freely and quickly: see Figures 5–7 for some typical frames. No special lighting was used, and no make-up was applied to enhance the facial features. A range of facial expressions is represented, and the face is

[1] At first sight there seems little point in sampling the image stream any faster than 50Hz, the video field rate. However, the tracking algorithm is of an iterative nature and a high tracking rate allows the algorithm to converge on the same image, improving its performance considerably.

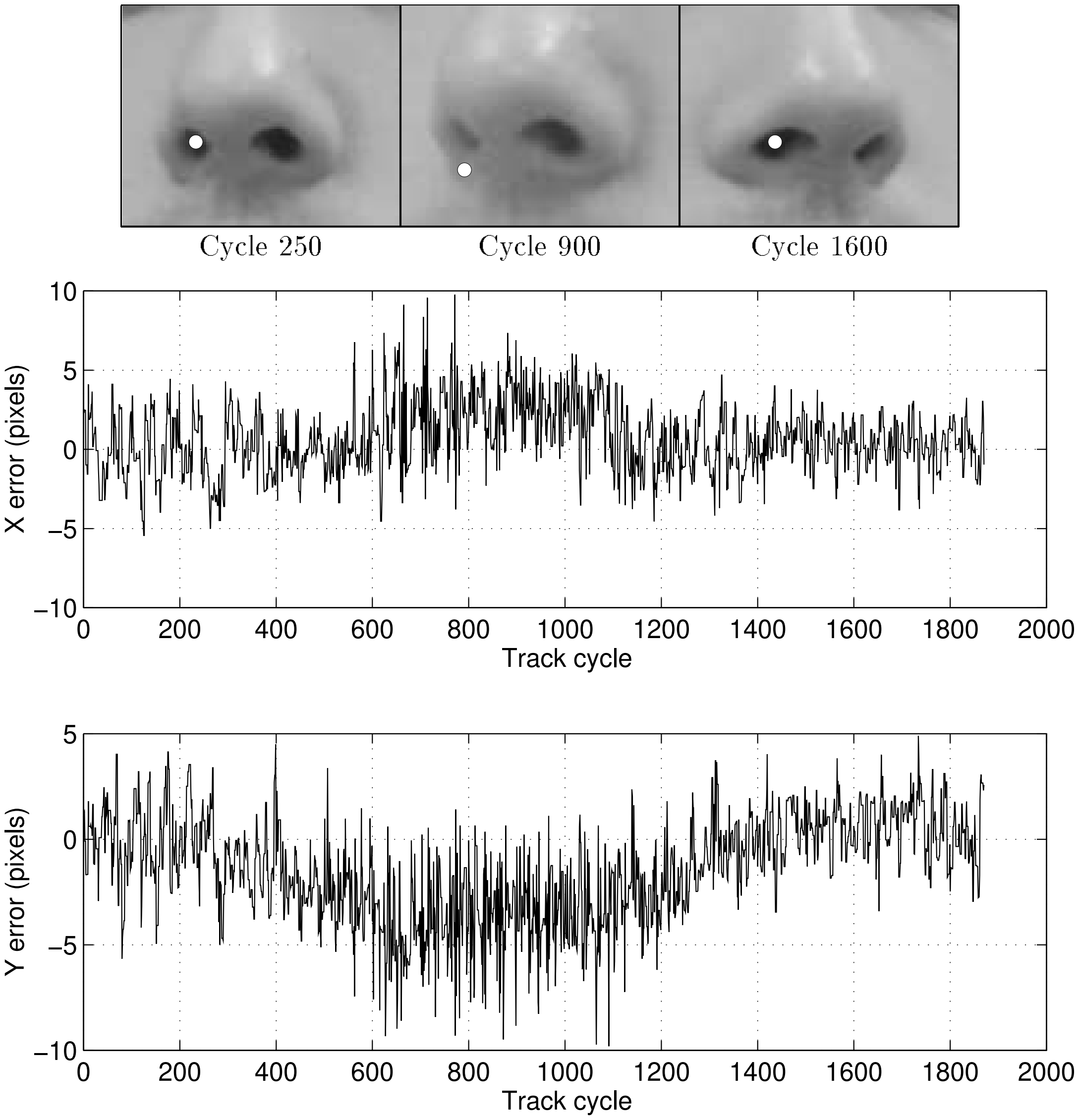


Figure 3: Tracking error of a typical feature detector.

To detect a nostril, a simple feature detector searches a 10×10 *pixel window around the expected nostril position and locates the darkest pixel. This simple strategy works well when the nostril is visible, finding the nostril position to within, typically, three pixels (which is reasonable, since the nostril measures about* 6×6 *pixels in the image). When the head is turned so that the nostril is less prominent, between cycles 500 and 1100, the feature detector starts to show a bias as it finds instead the shadow around the outline of the nose. There are also many isolated outliers (gross errors) in the* x *and* y *traces. This sort of behaviour is typical of simple visual feature detectors, and demonstrates that it is unwise to assume zero-mean Gaussian noise on the detector outputs. The experiment is slightly flawed in that the "true" nostril position is obtained by back-projecting the output of a visual tracker, which is of course not perfect, but the non-stationary noise processes are nevertheless evident.*

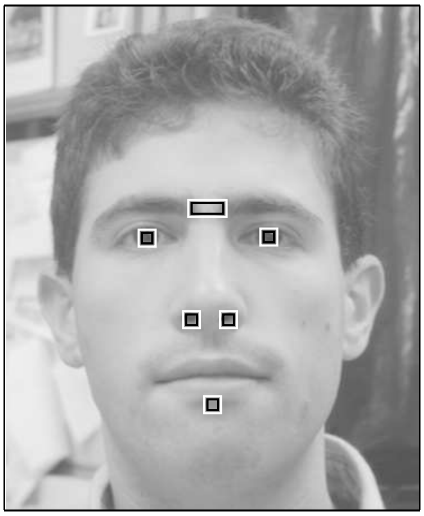


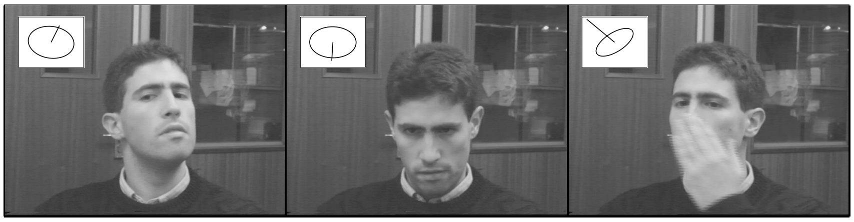
Figure 4: Features and search windows for face tracking.

Dark-pixel finders (which simply return the location of the darkest pixel within their local search windows) were used to detect the eyes, nostrils and the shadow below the lower lip. A coarse correlation-based detector was used to detect the centre of the eyebrows using a dark-light-dark correlation template.

partially occluded about 20 seconds into the sequence. The motion is fairly unconstrained, with angular velocities up to 1.4 rad/s, and angular accelerations up to 14 rad/s^2.

Part of the output of the tracker (the orientation of the face) is shown in Figure 5. Also displayed is the magnitude of the implied angular acceleration, which reflects how smooth the tracking is. The first thing to notice is that the pose estimates are a little noisy. This is because the tracker is operating with a relatively small number of data points, six. In [4] it is estimated that a consensus set of eight points provides a 95% probability that compatibility with an incorrect model will not occur. So even with eight points, an erroneous pose might explain *all* the data points every twenty or so frames. With six points, we would expect a fairly noisy output, as erroneous poses frequently admit sizable consensus sets. This effect is most pronounced when the feature detectors are least reliable, and in fact the face is lost about 20 seconds into the video sequence, which is when the hand occludes part of the face.

There are two immediate steps which could be taken to improve the tracker. We could include more points in the model, but this would slow the tracker and there is of course no guarantee that any more suitable features can be found on the target. Alternatively, we could assume smooth motion and filter the pose estimates over time. The traditional way to do this would be to attach a Kalman filter to the output of the tracker, as in [3, 7]. A more elegant approach, which we consider in the sequel, would be to combine the filtering with the pose estimation. In so doing we avoid the cumbersome machinery of the Kalman filter, with its computationally expensive matrix inversions. We can also assume a more appropriate measurement noise model than the Gaussian form implicit in the standard Kalman filter framework.



Cycle 815 Cycle 1320 Cycle 2200

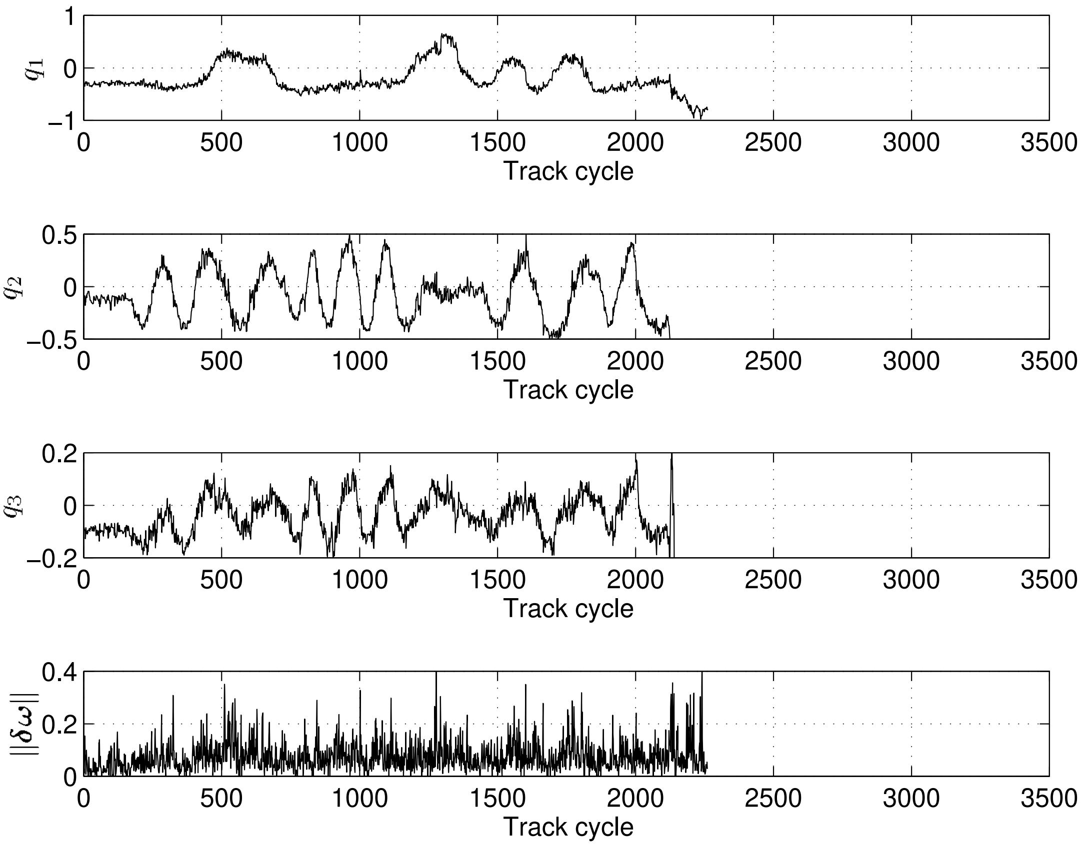


Figure 5: Performance of the consensus tracker.

The upper three traces show the estimated orientation of the face in an angle-axis form: the axis of rotation is given by (q_1, q_2, q_3)*, and the magnitude of the rotation is* $\|\mathbf{q}\|$ *rads. The orientation is illustrated as a drawing pin in some sample frames at the top of the figure. The bottom trace shows the motion smoothness in terms of the magnitude of the angular acceleration* $\|\boldsymbol{\delta\omega}\|$*, given in rads/cycle*^{2}*. Tracking is maintained until about 20 seconds into the sequence, when a hand partially obscures the face.*

A temporal continuity tracker

The RANSAC/alignment tracker can be modified so that the candidate poses are selected not on the sizes of their consensus sets, but on the smoothness of the implied motion. More formally, we replace step 3 of the algorithm with:

> Repeat 1 and 2 for all combinations of three points. Choose the pose that produces the smoothest motion based on previous frames.

With high sampling rates and typical facial motions, we expect little motion from frame to frame. If we assume zero-mean Gaussian distributions for the linear and angular velocities, we could select the pose which minimizes

$$\exp\left(-\frac{\|\mathbf{v}\|^2}{2\sigma_l^2} - \frac{\|\boldsymbol{\omega}\|^2}{2\sigma_a^2}\right) \tag{2}$$

where $\mathbf{v}$ is the linear velocity implied by the new pose, $\boldsymbol{\omega}$ is the angular velocity, and σ_l and σ_a are the standard deviations of $\|\mathbf{v}\|$ and $\|\boldsymbol{\omega}\|$ respectively. It is not necessary to attempt to model the individual components of velocity and their covariances. If the sampling rate was lower and the motion faster, then a constant velocity model would be more appropriate than a zero velocity one: however, the resulting system would be more sensitive to noise and poor predictions could lead to complete tracking failures, as in Figure 1. When the motion from frame to frame is small enough, a zero velocity model imparts greater robustness to the overall tracking system.

A tracker was built using this algorithm and applied to the standard video sequence of the moving head: the results are shown in Figure 6. The bottom trace indicates significantly smoother tracking, and the quality of the output would be acceptable for most applications. However, the tracker loses lock around cycle 850, where the head executes a high velocity manoeuvre. This is not too surprising: the true pose implied a large velocity which would not have scored well on the basis of (2). The tracker is over-smoothed, behaving in much the same way as a more conventional Kalman-filtered tracker set up with too much confidence in the motion model and too little confidence in the measurements.

A temporal consensus tracker

The key to successful tracking is to make use of both the intra-frame consensus information and the inter-frame motion constraints in a sound, coherent manner. This can be accomplished within a Bayesian inference framework. Suppose we have a set of measurements m and a pose hypothesis $\mathcal{H}$. Then

$$P(\mathcal{H} \mid m) = \frac{P(m \mid \mathcal{H})\, P(\mathcal{H})}{P(m)} \tag{3}$$

Our ultimate goal is to maximise the $P(\mathcal{H}|m)$ term, the *a-posteriori* probability. This term depends on two probability functions. The $P(\mathcal{H})$ term, the *prior*, expresses our prior belief in a particular pose in the absence of any measurements. In this case it would embody the smooth motion assumption. The $P(m \mid \mathcal{H})$ term, the *evidence*, should reflect the performance of the feature detectors. The $P(m)$ term is merely a normalization function, set to ensure that $P(\mathcal{H}\,|\,m)+P(\neg\mathcal{H}\,|\,m) = 1$. Direct maximization of the a-posteriori probability is infeasible: for most realistic evidence and prior terms, the a-posteriori probability is a

Cycle 550 Cycle 815

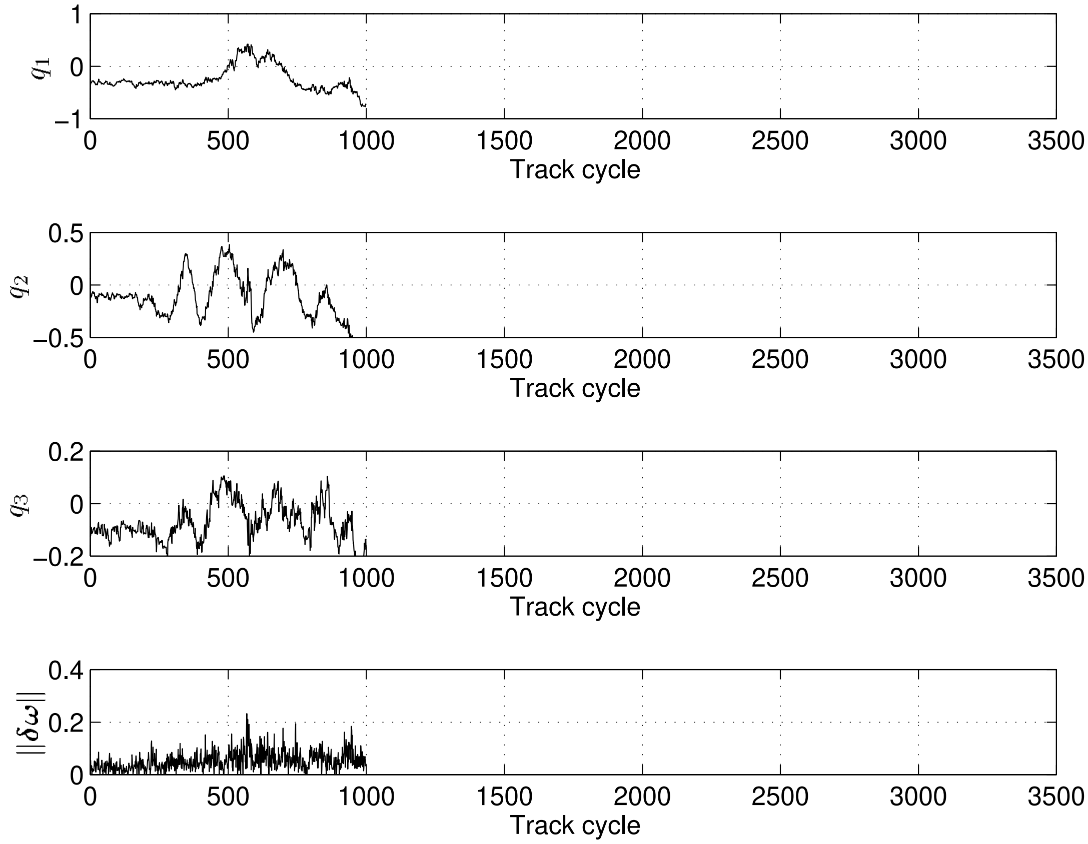


Figure 6: Performance of the temporal continuity tracker.

Since the motion in the image sequence is predominantly rotational, the tracker was implemented using the Gaussian distribution (2) with no $\|\mathbf{v}\|$ *term. Tracking is maintained until the face executes a high velocity manoeuvre around cycle 800. The bottom trace indicates significantly smoother tracking than was achieved with the consensus tracker.*

complicated function which would resist analytical or fast numerical maximization. But we could use the a-posteriori probability to choose between the candidate pose solutions produced by the RANSAC/alignment algorithm.

As with all such inference processes, the particular choice of a prior probability function is somewhat arbitrary, so long as it reflects any knowledge we have about the target's motion. If we assume that the magnitudes of the target's linear and angular velocities are Gaussian distributed with zero mean and standard deviations σ_l and σ_a respectively, then a suitable prior would be[2]

$$P(\mathcal{H}) \propto \exp\left(-\frac{\|\mathbf{v}\|^2}{2\sigma_l^2} - \frac{\|\boldsymbol{\omega}\|^2}{2\sigma_a^2}\right) \tag{4}$$

where $\mathbf{v}$ is the linear velocity implied by the new pose, and $\boldsymbol{\omega}$ is the angular velocity.

We could also use a Gaussian function for the evidence term, reflecting noisy feature detectors with zero bias and standard deviation σ_d pixels. If d_i is the discrepancy in pixels between an observed and back-projected feature location, then

$$P(m \mid \mathcal{H}) \propto \exp\left(\frac{-\sum_i d_i^2}{2\sigma_d^2}\right) \tag{5}$$

However, with such an evidence function a rogue measurement could still over-influence the selection process. More in keeping with the philosophy of this paper would be to use a discrete measurement model which describes the probability p that each detector locates a feature to within r pixels of its true position. Then, considering the n features which were *not* used to calculate the pose, if m of them are detected within r pixels of their back-projected image positions, we have

$$P(m \mid \mathcal{H}) = \, ^nC_m \; p^m (1-p)^{n-m} \tag{6}$$

We are, in effect, assigning a probability distribution to the size of the RANSAC consensus set. With this Binomial distribution, a distant outlier would influence the selection process to the same extent as a mild outlier, reflecting the fact that the feature detectors are about as likely to produce a gross error as they are a moderate one.

Formally, the temporal consensus tracker is identical to the consensus tracker but with a modified step 3:

> Repeat 1 and 2 for all combinations of three points. Choose the pose that maximises
>
> $$P(\mathcal{H} \mid m) \propto \, ^nC_m \; p^m (1-p)^{n-m} \times \exp\left(-\frac{\|\mathbf{v}\|^2}{2\sigma_l^2} - \frac{\|\boldsymbol{\omega}\|^2}{2\sigma_a^2}\right) \tag{7}$$

The performance of the temporal consensus tracker, illustrated in Figure 7, is very good. The pose estimates show much of the smoothness of the temporal continuity tracker, without being over-smoothed: as a result, the tracker can cope with the high angular velocity around cycle 800, and the face is successfully tracked to the end of the 30 second

[2] For a continuous distribution of pose hypotheses, $P(\mathcal{H})$ is technically zero for any particular pose. To make our probability analysis rigorous, we must imagine a finely quantized hypothesis space and write $P(\mathcal{H}) \approx f_{\mathcal{H}}(\mathcal{H})\delta\mathcal{H}$, where $f_{\mathcal{H}}(\mathcal{H})$ is the probability density of $\mathcal{H}$ and $\delta\mathcal{H}$ is the size of the quantization bins. In the sequel, we shall abbreviate this notation and simply write $P(\mathcal{H}) \propto f_{\mathcal{H}}(\mathcal{H})$.

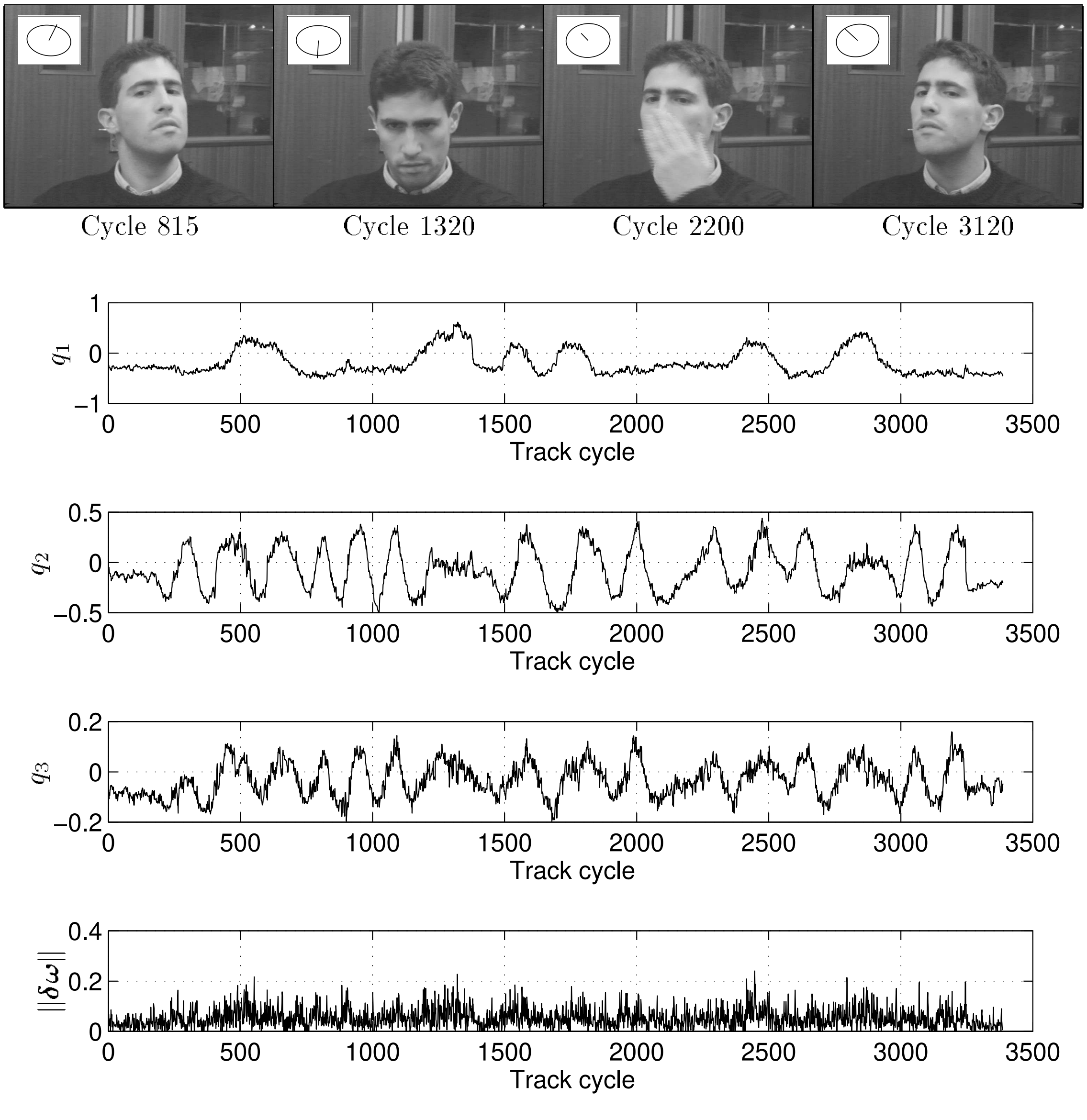


Figure 7: Performance of the temporal consensus tracker.

As with the temporal continuity tracker, no $\|\mathbf{v}\|$ *term was used in (7). The other parameters were set to reflect the nature of the motion and the feature detectors:* $\sigma_a = 0.04$ *rad/cycle (about twice the maximum angular velocity encountered in the image sequence) and* $p = 0.7$ *(the feature detectors get it right about 70% of the time). By properly combining single frame measurements and inter-frame constraints, tracking is maintained through high velocity manoeuvres and partial occlusion of the target. Moreover, the bottom trace indicates significantly smoother tracking than was achieved with the consensus tracker.*

Figure 8: Coping with different facial expressions.

The subject's face is being tracked with five dark-pixel detectors, locating the eyes, nostrils and the shadow beneath the lower lip. The tracker's orientation estimate is displayed as a drawing pin in the top left hand corner of the frame, and the back-projected feature locations are superimposed on the image. When the subject speaks and opens his mouth, teeth are visible where the lip shadow is expected to be and the model is no longer accurate. The tracker, however, can cope with this: it simply ignores the affected lip measurements, since they don't agree with other measurements or imply smooth motion.

video sequence. The tracker's success is remarkable given the lack of high contrast features on the target and the simplicity of the feature detectors. Moreover, the tracker can cope with different facial expressions (equivalent to nonrigidity of the target, producing slight model incompatibility) and mild occlusion of the face, since measurements of the affected features will not contribute to extended consensus sets or imply smooth motion: the a-posteriori probability function (7) will automatically select reliable features — see Figure 8. The temporal consensus tracker would appear to outperform other face trackers described in the literature [1, 12].

There is no reason why the tracker should not be equally successful with other types of target. It is only necessary to have a 3D model of, say, six or more detectable features on the target, and suitable, fast feature detectors. These features need not be points, since the alignment algorithm can be adapted to work with a mixture of points and lines [14]. With larger models it may not be possible to evaluate all the 3-point combinations and still achieve frame-rate operation. However, the tracking speed could be maintained using random sampling, in the spirit of the original RANSAC paradigm.

4 Conclusions

A new model-based tracking algorithm has been proposed, eschewing least-squares pose fitting in favour of a more robust minimal information approach. The well-established "alignment" algorithm has been embedded within the RANSAC regression paradigm to form the heart of the tracker. By elegantly combining intra-frame observations with inter-frame motion constraints, it is possible to avoid a separate and costly filtering stage: moreover, the inappropriate Gaussian observation noise assumption is replaced by a more realistic Binomial model. The resulting tracker produces state-of-the-art performance when applied to the difficult task of tracking a human face under adverse conditions, and should be equally successful with a variety of other targets.

Acknowledgements

Andrew Gee gratefully acknowledges the financial support of Queens' College, Cambridge, where he is a Research Fellow. Thanks are also due to Kin Choong Yow, for agreeing to be a model in some of the face tracking experiments.

A Model acquisition

Given enough model points to maintain tracking under controlled conditions (four will usually do if the target is unoccluded and moving slowly), it is straightforward to acquire new model points using a Kalman filter [5]. The filter's state variable, $\mathbf{x}_k$, is the current estimate of the new feature's 3D model-centered coordinates. The filter also maintains an estimate of the state vector's covariance matrix P_k.

The filter is initialized with a rough guess of the model coordinates $\mathbf{x}_0$, and a covariance matrix P_0 with large entries to reflect an initial lack of confidence in the state vector. Then the target is tracked using the temporal consensus tracker and the established model, while the image position of the new feature is followed using an appropriate feature detector. Measurements are taken of the new feature's observed image location minus the scaled image plane translation:

$$\mathbf{z}_k = \begin{bmatrix} x \\ y \end{bmatrix} - s \begin{bmatrix} t_x \\ t_y \end{bmatrix}$$

From (1), the expected value of this measurement is given by $H\mathbf{x}_k$, where

$$H = s \begin{bmatrix} R_{11} & R_{12} & R_{13} \\ R_{21} & R_{22} & R_{23} \end{bmatrix}$$

The difference between the observation and the expectation is termed the innovation, $\mathbf{v}_k$. The variance associated with the innovation is given by

$$S_k = E\left[\mathbf{v}_k \mathbf{v}_k^T\right] = H P_k H^T + R$$

where R is the variance of the observation noise. The state vector and covariance matrix are updated as follows:

$$\begin{aligned} \mathbf{x}_{k+1} &= \mathbf{x}_k + W_k \mathbf{v}_k \\ P_{k+1} &= P_k - W_k S_k W_k^T \\ \text{where } W_k &= P_k H^T S_k^{-1} \end{aligned}$$

Figure 9 shows the Kalman filter converging as a new feature, in this case a small mole, is acquired on a moving face. The model-centered coordinates of the mole are established within a few seconds.

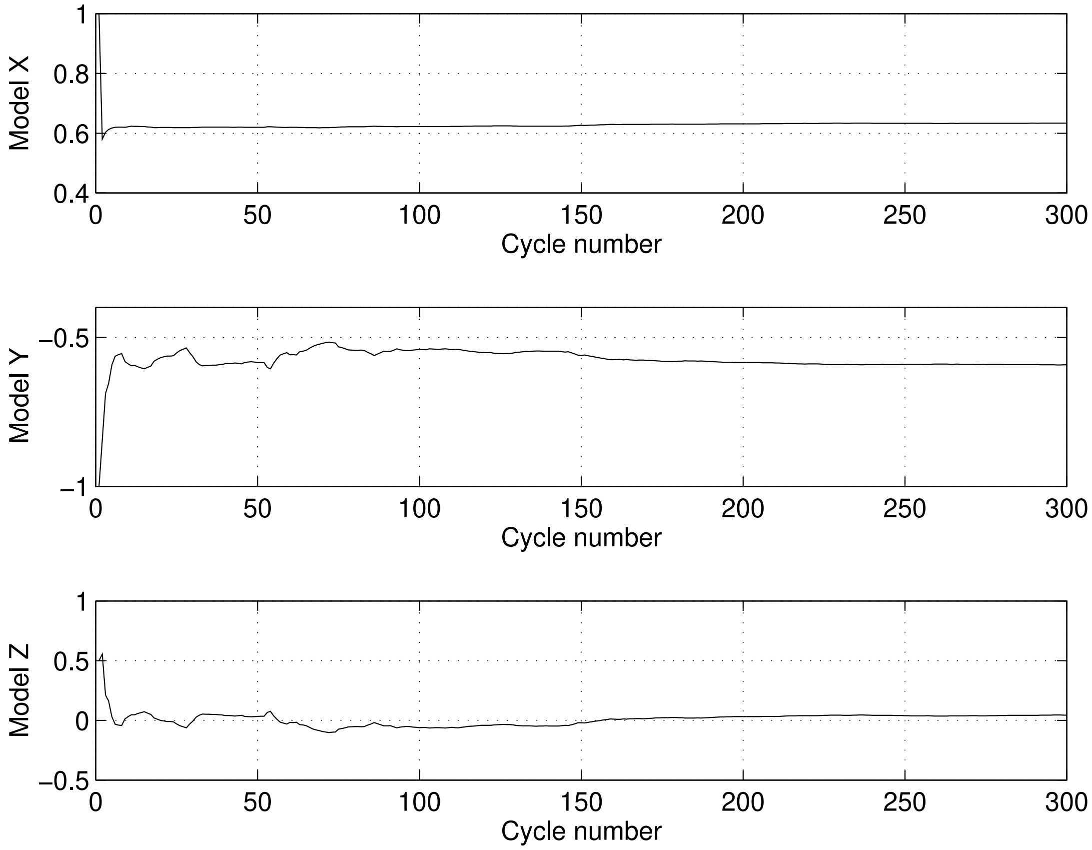


Figure 9: Acquiring the 3D model-centered coordinates of a new feature.

The 3D model-centered coordinates of a small mole on a face are to be established. Starting with a rough guess of the coordinates, in this case $(1.0, -1.0, 0.5)$*, the mole is followed using a simple dark-pixel finder while the rest of the face is tracked using a temporal consensus tracker. The discrepancy between the mole's observed and expected image positions is fed into a Kalman filter, which updates the estimate of the 3D coordinates. Within a few seconds the mole's location on the face is accurately established at* $(0.63, -0.59, 0.05)$.

References

[1] A. Azarbayejani, T. Starner, B. Horowitz, and A. Pentland. Visually controlled graphics. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(6):602–605, 1993.

[2] R. Cipolla, Y. Okamoto, and Y. Kuno. Robust structure from motion using motion parallax. In *Proceedings of the Fourth International Conference on Computer Vision*, pages 374–382, Berlin, 1993.

[3] R. Evans. Kalman filtering of the pose estimates in applications of the RAPID video rate tracker. In *Proceedings of the British Machine Vision Conference*, pages 79–84, Oxford, 1990.

[4] M. A. Fischler and R. C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981.

[5] A. Gelb, editor. *Applied Optimal Estimation.* MIT Press, Cambridge MA, 1974.

[6] D. Gennery. Visual tracking of known three-dimensional objects. *International Journal of Computer Vision*, 7(3):243–270, 1992.

[7] C. Harris. Tracking with rigid models. In A. Blake and A. Yuille, editors, *Active Vision*, pages 59–73. MIT Press, Cambridge MA, 1992.

[8] N. J. Hollinghurst and R. Cipolla. Uncalibrated stereo hand-eye coordination. *Image and Vision Computing*, 12(3):187–192, 1994.

[9] R. Horaud, B. Conio, and O. Leboulleux. An analytic solution for the perspective 4-point problem. *Computer Vision, Graphics, and Image Processing*, 47:33–44, 1989.

[10] D. P. Huttenlocher and S. Ullman. Recognizing solid objects by alignment with an image. *International Journal of Computer Vision*, 5(2):195–212, 1990.

[11] D. Lowe. Robust model-based motion tracking through the integration of search and estimation. *International Journal of Computer Vision*, 8(2):113–122, 1992.

[12] K. Mase, Y. Watanabe, and Y. Suenaga. A realtime head motion detection system. *SPIE Volume 1260: Sensing and Reconstruction of Three-Dimensional Objects and Scenes*, pages 262–269, 1990.

[13] P. Meer, D. Mintz, A. Rosenfeld, and D. Y. Kim. Robust regression methods for computer vision: A review. *International Journal of Computer Vision*, 6(1):59–70, 1991.

[14] D. Shoham and S. Ullman. Aligning a model to an image using minimal information. In *Proceedings of the Second International Conference on Computer Vision*, 1988.

[15] R. S. Stevens. Real-time 3D object tracking. *Image and Vision Computing*, 8(1):91–96, February 1990.

[16] D. W. Thompson and J. L. Mundy. Model-based motion analysis — motion from motion. In R. C. Bolles and B. Roth, editors, *Robotics Research: The Fourth International Symposium*, pages 299–309. MIT Press, Cambridge MA, 1988.