*British Machine Vision Conference-97, Essex, UK, September 1997(**selected for the special issue of Image and Vision Computing**)*.

Robust Recursive Structure and Motion Recovery under Affine Projection

Miroslav Trajković and Mark Hedley
Department of Electrical Engineering
University of Sydney, NSW 2006, Australia
miroslav@ee.usyd.edu.au

Abstract

In this paper we present an algorithm for structure and motion (SM) recovery under affine projection from video sequences. The algorithm tracks the motion of a single structure, be it an object or the entire scene itself, allowing for any type of camera motion. This could be used for example to track the motion of a vehicle in a warehouse (single object, static camera) or for visual navigation from a moving platform (track scene from moving camera). The algorithm requires a set of features to be detected in each frame, and that at least four features are correctly matched between each three consecutive frames. Compared to previous algorithms this novel algorithm has a lower computational cost, making it attractive for real time applications. Our algorithm also provides dynamical detection of outliers and allows for previously lost features to reappear in the sequence.

1. Introduction

A structure and motion recovery is one of the fundamental problems in computer vision due to large spectrum of applications ranging from mobile robots navigation to multimedia and different tracking problems.

One of the most widely used approaches to the SM recovery is the factorization algorithm proposed by Tomasi and Kanade [9]. As an input, they used image projections of a number of features detected and tracked over several frames, and computed SM by the means of SVD. Their algorithm is developed for an orthographic camera, and was shown to be robust and accurate in the presence of Gaussian noise. Although useful as a starting point, their algorithm has several limitations in terms of real time processing. First, this is a batch method and can only compute the parameters once all measurements have been made. The algorithm is also computationally very expensive (due to the use of SVD), and does not deal easily with outliers, and missing and new features. The last can be a serious problem, especially if no single feature is available in all frames. Morita and Kanade [6] improved original factorization method, and developed a sequential version, that has almost same accuracy but much lower computational burden. The computational cost of their algorithm is $O(k^2)$, compare to $O(nk^2)$ required by SVD, for k frames and n features. They addressed neither outlier detection nor the inclusion of new or removal of old features.

Held [1] has partially extended the Morita and Kanade algorithm to include new features. He assumed that image stream consists of several overlapping sequences, each consisting of 40 to 50 frames, that during each sequence same number of features has been tracked, and that every two consecutive sequences share some (unspecified by author, but at least four according to [2]) number of points. He computed motion and structure parameters in each sequence separately and used the “shared” points to relate parameters between each two consecutive sequences. However, the assumption that all (or at least, many) features are tracked over 40-50 frames, sounds a bit unrealistic from our experience with existing corner detectors. The algorithm has been verified on synthetic measurements only, with added Gaussian noise. McLauchlan *et al* [3,4] used the variable state-dimension filter (VSDF) to handle missing and new features. They posed the structure from motion problem as a parameter estimation problem and solved it by using the Extended Kalman Filter (EKF). They updated the complete structure and *only last* motion estimate at each time step and therefore achieved low computational cost $O(k)$ compare to when complete motion matrix has been updated. The authors have mentioned outlier rejection, but they gave neither a method nor a reference. They did not give an algorithm to deal with the reappearance of features.

In this paper we present an algorithm that iteratively computes the structure and motion parameters at each time step. It has the same computational cost $O(n+k)$ as McLauchlan *et al*, but it updates both the structure and the complete motion parameters, so it converges to the true solution. It is made possible by noting that structure should remain constant over time and by updating it at each time step. Motivated by work of Reid and Murray [7] we have extended their algorithm so that it can easily include new features and recover the complete structure and motion provided that in each set of three consecutive frames there are at least four successfully matched corners. This is a reasonable condition in practice and is almost always fulfilled for almost any corner detector. Further, we have detected outliers at each time step, as points which are inconsistent with recovered motion model, or more precisely, as points that have a large influence on the estimated parameters and their standard variation. Finally we have developed a procedure that checks whether newly appeared features are genuinely new, or old features that have reappeared.

2. Problem Statement

In this paper we assume an affine camera model which has the form:

$$p = MS + t \tag{1}$$

where S is 3×1 is the world coordinate of a feature point, p is 2×1 image projection of S, M is 2×3 projection matrix and t is 2×1 translation vector. This model is a generalisation of orthographic camera model [6,9] and is a good approximation of perspective projection when change in depth is small comparing to average distance from the optical center and this condition is almost always satisfied for independently moving objects. If the object of interest undergoes rigid motion the image projections will not change if we fix the world coordinates of the object of interest and change camera parameters M, t accordingly. Hence, without loss of generality, we may assume that object of interest is static and that the camera is moving.
Given k projections (images) of n scene points we want to recover camera motion parameters $M(j)$, $t(j)$, $j = 1,2,\cdots,k$ and structure parameters S_i, $i = 1,2,\cdots,n$ (world vectors of tracked scene points). These parameters are related by the measurement equation:

$$p_i(j) = M(j)S_i + t(j) \tag{2}$$

where $p_i(j)$ is the projection of i^{th} point onto the j^{th} image, and i and j vary from $1\ldots n$ and $1\ldots k$ respectively. To compute the SM parameters using all the measurement equations, one must minimise the objective function:

$$C(k) = \sum_{j=1}^{k}\sum_{i=1}^{n}\left\|p_i(j) - M(j)S_i - t(j)\right\|^2 \tag{3}$$

but for the real time implementation the parameters must be computed following the acquisition of each frame. It may be shown that the translational component $t(j)$ at each frame is given by the centroid of the feature locations $t(j) = \overline{p_i(j)}$ and equation (2) can now be rewritten as $w_i(j) = M(j)S_i$ where $w_i(j) = p_i(j) - t(j)$, or in matrix form:

$$W = MS \tag{4}$$

where $W = [w_i(j)]_{k\times n}$, $M = [M(1)^T \cdots M(k)^T]^T$ and $S = [S_1 \cdots S_n]$, and this is the equation obtained (although in different way) by Tomasi and Kanade [9]. To solve it, they employed singular value decomposition of measurement matrix and shown that $M = U_3 \Sigma_3^{1/2}$ and $S = \Sigma_3^{1/2} V_3^T$,where U_3, Σ_3 and V_3 are submatrices of *U*, *Σ* and *V* corresponding to the three largest singular values of *W*. In addition, motion and structure can be determined separately without computing the full SVD, since

$$W^T W = V \Sigma^2 V^T \quad \text{and} \quad WW^T = U \Sigma^2 U^T \tag{5}$$

and therefore the structure and motion are given by the eigendecompositions of $W^T W$ and WW^T respectively.

Note that equation (4) does not have unique solution. In fact, if *A* is an arbitrary invertible 3×3 matrix the matrices *MA* and $A^{-1}S$ are valid solution [9]. To ensure unique solution, we fixed first two rows of *M*, and now in equation (3) *j* will range from 2…*k*.

3. COMPLETE ALGORITHM

3.1 Recursive structure and motion recovery

Unlike Morita and Kanade who have computed motion and structure by iteratively updating equation (4) and McLauchlan who applied EKF to the measurement equation (2), we have employed direct minimisation of the cost function (3). For some initial number of frames (usually 3) we have initialised motion and structure matrices by solving equation (4). After a new frame has been acquired the cost function is iteratively minimised taking previously computed structure and motion parameters as initial values. We have developed two methods, and they will be explained here in more details.

First, let us suppose that cost function *C*(*k*) has been minimised, *i.e.* that the values *M*, *t* and *S*(*k*) are known. Our goal is to minimise cost function

$$C(k+1) = C = \sum_{j=2}^{k+1} \sum_{i=1}^{n} w_{ij} \left\| p_i(j) - M(j)S_i - t(j) \right\|^2 \tag{6}$$

where w_{ij} is a binary weight which expresses presence (w_{ij} = 1) or absence (w_{ij} = 0) of i^{th} feature in j^{th} frame. Since it does not effect the rest of derivation it will be omitted for the clarity. Equation (5) may be rewritten in more concise form as:

$$C = \sum_{j=2}^{k+1} \sum_{i=1}^{n} \left\| p_i(j) - M(j)S_i - t(j) \right\|^2 = \sum_{j=2}^{k+1} \sum_{i=1}^{n} \left\| p_i(j) - h_i(j) \right\|^2 = \sum_{j=2}^{k+1} \sum_{i=1}^{n} \left\| v_i(j) \right\|^2 \tag{7}$$

It will be seen later, that it is helpful to group motion parameters into the motion vector rather than using *M* and *t* separately and we define the motion vector as $m = \begin{bmatrix} M_{11} & M_{12} & M_{13} & t_1 & M_{21} & M_{22} & M_{23} & t_2 \end{bmatrix}$. To minimise the cost function we differentiate (6) to obtain

$$\frac{\partial C}{\partial m(j)} = \sum_{i=1}^{n} D_i^T v_i(j) = 0, \qquad j = 2,3,\dots,k+1$$
$$\frac{\partial C}{\partial S_i} = \sum_{j=2}^{k+1} E^T(j) v_i(j) = 0, \qquad i = 1,2,\dots,n \tag{8}$$

where

$$D_i = \frac{\partial h_i(j)}{\partial m(j)} = -\frac{\partial v_i(j)}{\partial m(j)} = \begin{bmatrix} X_i & Y_i & Z_i & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & X_i & Y_i & Z_i & 1 \end{bmatrix}$$
$$E(j) = \frac{\partial h_i(j)}{\partial S_i} = -\frac{\partial v_i(j)}{\partial S_i} = \begin{bmatrix} M_{11}(j) & M_{12}(j) & M_{13}(j) \\ M_{21}(j) & M_{22}(j) & M_{23}(j) \end{bmatrix}.$$

There are three different ways to solve this equation, and we will show them and give some comparison.

1) Linearizing (7) around previously estimated structure and motion and assuming that D_i and $E(j)$ are constant. Since

$$v_i(j) = \hat{v}_{ij} - D_i \Delta m(j) - E^T(j) \Delta S_i$$

equation (7) can be written as the linear system of $n+k+1$ equations

$$A(j)\Delta m(j) + \sum_{i=1}^{n} B_{ij} \Delta P_i = \sum_{i=1}^{n} D_i^T \hat{v}_{ij}, \qquad j = 2,3,\dots,k+1$$
$$\sum_{j=2}^{k+1} B_{ij}^T \Delta m(j) + C_i \Delta P_i = \sum_{j=2}^{k+1} E^T(j) \hat{v}_{ij} \qquad i = 1,2,\dots,n \tag{9}$$

where

$$A(j) = \sum_{i=1}^{n} D_i^T D_i, \quad B_{ij} = D_i^T E(j), \quad C_i = \sum_{j=2}^{k+1} E^T(j) E(j) \tag{10}$$

and this is the same set of equations as obtained in ML2. The authors have shown that this set can be solved in $o(\min(n^3, k^3))$ computational time, because of some decomposable properties (otherwise the computational load would be $o((n+k)^3)$). We have investigated convergence properties of this algorithm and have shown that this algorithm has linear rate of convergence (the proof is omitted due to lack of space).

2) Linearizing equation (7) around the previously found structure and motion parameters. In this case we linearize not only $v_i(j)$ but the whole expressions. Interestingly, we obtained a set of linear equations of the same form as (8), the only difference being that

$$B_{ij} = D_i^T E(j) - F_{ij}, \qquad F_{ij} = \frac{\partial D_i^T}{\partial S(j)} v_i(j).$$

Clearly, computational cost remains the same (the cost of computing F_{ij} is negligible), but because of this correcting part, the rate of convergence is quadratic.

3) Applying the method of simple iteration; In terms of m and S, $h_i(j)$ can be written as:

$$h_i(j) = D_i m(j) = E(j) S_i$$

and thus, equation (7) can be rewritten as

$$\sum_{i=1}^{n} D_i^T (p_i(j) - D_i m(j)) = 0, \qquad j = 2,3,\ldots,k+1$$
$$\sum_{j=2}^{k+1} E^T(j)(p_i(j) - E(j)S_i) = 0, \qquad i = 1,2,\ldots,n$$

or,

$$m(j) = A(j)^{-1} \sum_{i=1}^{n} D_i^T p_i(j), \qquad j = 2,3,\ldots,k+1 \tag{11a}$$

$$S_i \quad = C_i^{-1} \sum_{j=2}^{k+1} E^T(j) p_i(j), \qquad i = 1,2,\ldots,n \tag{11b}$$

where $A(j)$ and C_i are defined in (10). This method also have linear convergence, but computational cost is $O(n+k)$. We have also found that this method has more stable convergence for small number of features (5 to 15) than previous two, and that in practice three to five iterations suffices.

Initialisation

To perform any of above mentioned algorithms, we need initial estimates of structure and motion parameters. Since structure should remain constant over the time, previously estimated structure matrix S is usually quite good initial guess. It also holds. It also holds for motion vectors $m(1),\ldots, m(k)$. To estimate $m(k+1)$, equation (10a) can be used and usually quite good estimate is made.

3.2 Matching and outlier rejection

In order to obtain measurement matrix we have to correctly match corners from C_1 to C_2, from C_2 to C_3, … etc., where $C_1, C_2, \ldots, C_n$ refers to the set of corners in images $I_1, I_2, \ldots, I_n$ respectively. Usually, matching two sets of corners involves the use of cross correlation between small patches around each corner and this has been described elsewhere [10]. This procedure, however, will almost always give some (usually) small number of incorrect matches (outliers), which can (and usually will!) significantly bias structure and motion estimates. It is therefore necessary to detect incorrect matches and disregard them when solving for structure and motion. For this ask, we have developed a two steps technique, based on the use of robust statistics, concretely, we have employed Least Median of Squares (LMedS) estimator which will work when number of outliers is less 50% of total points, which is reasonable assumption and almost always fulfilled in practice. For more details about LMedS interested reader is referred to [5].

1) *Matching between two images.*

In the first step, we match the corners from C_1 to C_2 and obtain the measurement matrix:

$$W_{12} = \begin{bmatrix} x_1^{(1)} & \cdots & x_n^{(1)} \\ y_1^{(1)} & \cdots & y_n^{(1)} \\ x_1^{(2)} & \cdots & x_n^{(2)} \\ y_1^{(2)} & \cdots & y_n^{(2)} \end{bmatrix}$$

Now, we randomly select four columns and obtain sample matrix

$$W_S = \begin{bmatrix} x_{i1}^{(1)} & \cdots & x_{i4}^{(1)} \\ y_{i1}^{(1)} & \cdots & y_{i4}^{(1)} \\ x_{i1}^{(2)} & \cdots & x_{i4}^{(2)} \\ y_{i1}^{(2)} & \cdots & y_{i4}^{(2)} \end{bmatrix}$$

which we decompose using SVD.

$$W_S = M_S S_S + t_S .$$

(Actually, since we only need M_S and t_S we can exploit (5). Having found motion parameters M_S and t_S, we find structure vector of the remaining matched corner points C_R solving equation

$$W_R = M_S S_R + t_S ,$$

where W_R is a measurement matrix corresponding to the C_R. S_R can be easily computed using pseudo inverses as $S_R = M_S^{\#}(W_R - t_S)$, where $M_S^{\#}$ is pseudo inverse of M_S. The difference between measured corner positions and positions which best fit the model (M_S, t_S) is given by

$$D = W_R - (M_S S_R + t_S)$$

and the total distance (error) between measured and predicted corner position of i^{th} point is $\varepsilon_i = \|d_i\|$, where d_i refers to i^{th} column of *D*. The standard variation of the error vector ε is computed using formula [5]

$$\sigma^0 = 1.4826\left(1 + \frac{5}{n-p}\right)\underset{i}{\text{median}}\left(\varepsilon_i\right)$$

where *n* is number of points in and *p* is the number of sampling points (4 in our case). If our sample consists only of inliers, due to use of median (which disregards outlier r

Finally, outliers are found as points which residual error is outside confidence interval 2.5σ. Motion parameters are now computed by solving equation (5) and taking into account inliers only.

With computed motion parameters in hand, we can establish affine epipolar geometry, *i.e.* for each point A in I_1 we can find epipolar line e_A in I_2 and *vice versa*. We now try to match unmatched corners from both images taking into account epipolar constraint (*i.e.* that match of point A must lie on its epipolar line e_A).

The whole procedure of corner matching is now repeated to match the corners from C_2 to C_3.

2) *Matching between three images*

By matching point from two images the rigidity constraint can not be exploited, and we can only use epipolar constraint. Epipolar constraint, although reducing searching space significantly and therefore reduce possibility of incorrect matches, still leaves ambiguity in matching, because all the point on the epipolar line of point A are geometrically correct matches.

This ambiguity can be avoided by using corner matching across three images. Namely, given position of the feature in any two images, and motion parameters M and t its position in the third image is uniquely determined. Hence, by using this procedure, we can detect those outliers which are "correctly" matched over each pair of consecutive images (satisfy epipolar constraint) but do not satisfy rigidity constraint over three frames. The procedure is essentially same as for two images, the only difference being that measurement matrix now has six rows.

3.3 Inclusion, reappearance and deletion of features

As we mentioned before the only assumption of our algorithm is that in each three consecutive frames we have at least four *correctly* matched points (it will work even if the points are not matched correctly although results may be a bit weird). The new points will be included into measurement matrix only if they appear and are matched in last three consecutive frames and this procedure will be explained here. Let us consider frames n-2, n-1, n and n+1, and let C_1 denote a set of features which are matched over frames n-2 to n, while C_2 denotes the set of features which have been matched ***only*** over frames n-1 to n+1. Features from C_1 may or may not be matched in the last frame, and it does not affect the algorithm - the only condition is that the total number of features that are detected over last three frames is *not less than four*.

Let P be an arbitrary feature from C_2 and let $p^{(n-1)}$ and $p^{(n)}$ denote its image projections in the corresponding frames. Since the motion (and structure) are estimated over first n frames, the world vector of point P may be estimated by solving set of equations:

$$\begin{bmatrix} p^{(n-1)} \\ p^{(n)} \end{bmatrix} = \begin{bmatrix} M^{(n-1)} \\ M^{(n)} \end{bmatrix} S + \begin{bmatrix} t^{(n-1)} \\ t^{(n)} \end{bmatrix}$$

the solution of which is:

$$S = \begin{bmatrix} M^{(n-1)} \\ M^{(n)} \end{bmatrix}^{\#} \begin{bmatrix} p^{(n-1)} - t^{(n-1)} \\ p^{(n)} - t^{(n)} \end{bmatrix}$$

where # denote pseudo-inverse of a matrix.

Hence, for all points in $(n+1)^{\text{th}}$ frame we have estimated initial structure parameters and we can perform iterative update of structure and motion as described in Section 2.

Before we add the point from C_2 to measurement matrix, and perform update, we first check whether this point is genuinely new or it corresponds to some previously tracked feature

that has not been detected over (at least) last three frames either due to imperfection of corner detector or due to occlusion. For this task the following technique has been developed.
Let P_i be an arbitrary feature which has been tracked up to frame *n*-2 (inclusive) and let P_l be an arbitrary feature from C_2. We can say that P_i and P_l represent the same feature if the world distance between them is small enough, *i.e.* if

$$d_{il} = \|\Delta S\| = \|S_i - S_l\| < \alpha$$

where α is unknown threshold yet to be determined.
While we do not have any idea what is *small* in the structure space (especially because it is unique up to affine transformation), we can say what is small in the image space, *i.e.* we can say that *if* the features are close in structure space than the distance between their projections in the image plane has to be small to (typically three to five pixels). Note that vice versa does not hold, *i.e.* even if the image projections of points are the same it merely means that that they lie on the same projection line, while the distance between them can be arbitrary. Therefore, the following criteria has been applied: *Points P_i and P_l refer to the same physical points if the distance between their image projections is less than given threshold (β) independently of the pose between them, and for any recovered camera model M(1),… M(k).*
Mathematically, this condition can be expressed

$$\max_{\Delta S, j}\left(\|p_i(j) - p_l(j)\|\right) \le \beta \tag{12}$$

where

$$\begin{aligned} p_i(j) &= M(j)S_i + t(j) \\ p_l(j) &= M(j)S_l + t(j) \end{aligned}.$$

The difference between image projections is given by

$$\Delta p = p_i - p_l = M\Delta S$$

where index j has been dropped for the clarity. By applying Euclidean norm to both sides we obtain inequality relating distance between image projections to the distance between structure vectors

$$\|\Delta p\|_E \le \|M\|_S \|\Delta S\|_E$$

where index S denote spectral norm. Since the spectral norm is subordinate to Euclidean norm there exists at least one ΔS such that equality sign holds, and recalling (12) we get

$$\|\Delta p\|_E = \|M\|_S \|\Delta S\|_E \le \beta$$

and further

$$\|\Delta S\|_E \le \frac{\beta}{\|M\|_S}.$$

At this instance we just remind the reader that spectral norm of matrix M is defined as a square root of the maximum eigenvalue of the matrix $M^T M$, *i.e.* $\|M\|_S = \sqrt{\max \lambda(M^T M)}$. If $\lambda_j = \|M(j)\|_S$, than a threshold α is determined as

$$\alpha = \frac{\beta}{\lambda}$$

where $\lambda = \max_j \lambda_j$.

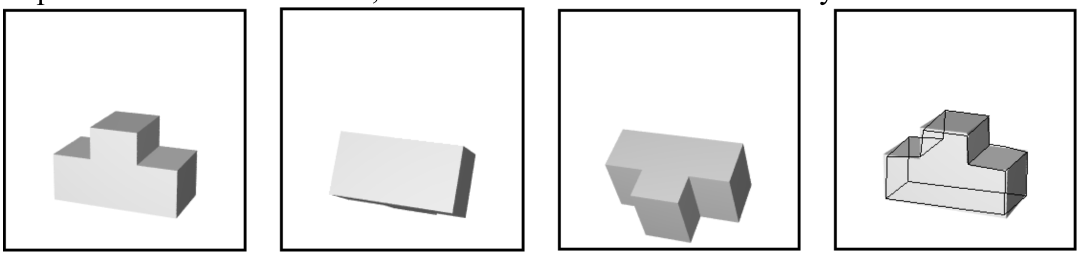
4. EXPERIMENTAL RESULTS

Our algorithm has been verified by testing on several video sequences.

POV image sequence

This sequence consists of 90 frames of an object rotating around the fixed axes (see Fig. 1 a–c). The rotation is a constant four degrees per frame. Corners are detected and matched using an algorithm similar to [10]. This is a difficult sequence, because no single corner appears in all the frames, and because the number of points is small - it varies from 6 to 13 per frame. For this reason, both algorithms proposed by Tomasi and Kanade [9] and Morita and Kanade [6] are not applicable for this sequence as the number of features is not constant. The VSDF algorithm may be used, but in its original form (without the reappearance of features) this will have a growing number of features as old features reappear, this will lead to slower computation and greater errors in the calculated parameters.

Using our algorithm we obtained total of 22 features, and by disregarding all of them which appeared in less than 20 frames we were left with total of 17 features which is quite close to the 16 distinguished corners on the object. Corners which have been effectively tracked over all the frames are shown on Fig 1d. As we can see they are all close to the true locations, including those which are currently occluded or not detected. Some distortion is present due to imperfection of affine model, but the structure has been correctly recovered.



a) b) c) d)

Fig. 1. POV sequence: 1st (a), 30th (b) and 60th (c) image in the sequence of 90 images. d) consistent features and recovered structure (the lines are shown for visualisation only).

Hotel Sequence

This sequence consists of 197 points tracked over 181 frames. Corners were detected in the first image and their positions in consequent images were determined using an optical flow technique [8] (see Fig. 2 a, b). The SM parameters were calculated using the SVD algorithm and our algorithm, which produced the same results, but the former took about an order of magnitude longer to compute. As in [4], we have found that updating motion parameters over all previous frames in each step is unnecessary, but unlike them,

a) b) c)

Fig.2 1st (a) and 181st (b) image from the hotel image sequence; c) Plots of the mean square error (cost function) with (lower curve) and without (upper curve) outlier rejection we performed multiple iterations for accuracy. A very important step here is the outlier detection. Since we have matches over all frames, the rigidity constraint were checked not over last three frames only, but over the whole sequence. The comparison of mean squared error with, and without outlier removal is given in Fig. 2c.

5. CONCLUSION

This paper addresses a practical problem in structure and motion recovery for real image sequences. The first problem is speed to allow real time operation, and we presented a fast batch recursive algorithm for structure and motion parameter computation and have shown that it gives the same results as Tomasi and Kanade method. Furthermore, we have developed an outlier detection technique that further decreased cost function (3). This paper is the first one (to our knowledge) that deals the problem of feature dropout and reappearance. We developed a procedure that allows previously lost features to reappear in the sequence and experimentally verified its operation.

ACKNOWLEDGMENTS

The hotel image sequence (along with corners and displacements) was provided by the Modeling by Videotaping group in the Robotics Institute, Carnegie Mellon University. Financial support was provided by the Australian Research Council.

REFERENCES

[1] Held A., "Piecewise Shape Reconstruction by incremental factorization", *Proc. 7th British Machine Vision Conference, vol I*, Edinburgh, UK, 1996, pp. 333-342.

[2] Koenderink J. J. and van Doorn A. J., "Affine structure from motion", *J. Opt. Soc. Am. A,* vol 8, pp. 377-385, 1991.

[3] McLauchlan P. *et al*. "Recursive affine structure and motion from image sequences" *Proc. 3rd ECCV, vol. I*, Stockholm, pp. 217-224, 1994.

[4] McLauchlan P. and Murray D.. "A unifying framework for structure and motion recovery from image sequences" *Proc. 5th ICCV*, MIT, pp. 314-320, 1995.

[5] Meer P. *et al*. "Robust regression methods for computer vision: a review", *Intl. Journal Computer Vision* vol. 6., pp. 59-70, 1991.

[6] Morita T. and Kanade T., "A sequential factorization method for recovering shape and motion from image streams", *Proc. 1994 ARPA Image Understanding Workshop, vol. II*, pp. 1177-1188, Monterey, CA1994.

[7] Reid I. D. and Murray D. W., "Active Tracking of Foveated Feature Clusters Using Affine Structure", *Intl. Journal Computer Vision,* vol. 18, pp. 41-60, 1996.

[8] Shi J. and Tomasi C., "Good Features to Track". *IEEE Conference on Computer Vision and Pattern Recognition*, June 1994, pp. 593-600.

[9] Tomasi C. and Kanade T., "Shape and motion from image streams under orthography: A factorization approach", *Intl. Journal Computer Vision,* vol. 9, pp. 137-154, 1991.

[10] Trajković M. and Hedley M., "Fast feature detection and matching for machine vision", *Proc. 7th British Machine Vision Conference, vol. I*, Edinburgh, UK, 1996, pp. 91-100.